THE MICROCOMPUTER MAGAZINE

commodore

JUNE/JULY, 1982

$2.50

THE MICROCOMPUTER MAGAZINE

commodore

FEATURES

DEPARTMENTS

commodore

Q&A HOTLINE

Q. I was pleased to read about the 64K expansion board. Will it accommodate the 32K (N) 2001 programs. If so, how are the programs displayed on the 80 column screen. Also, do you have a routine in BASIC so my Epson printer, when it has the auto line feed on, can print single line feeds instead of double line feeds.

H. Roth
Raleigh, NC

A. *The 64K expansion board can indeed load older version of BASIC. The add-on board program included with the board allows you to select the proper version and automatically sets the screen to 40 columns. It does this by using only 40 columns in the center of an 80 column screen. Most older programs can be run with little or no modification at all.*

Regarding your printer, the versions of BASIC previous to 4.0 sent a carriage/return and line/feed at the end of every PRINT# command. The way to suppress the line/feed is illustrated in this example:

```
PRINT#1,"THIS IS A TEST"
CHR$(13);
```

The semi-colon will suppress both the C/R and the L/F. The CHR$(13) has to be included to give the desired C/R.

Q. My machines says "### COMMODORE BASIC ###", and I understand this means I have Revision 3.0 ROM's. On my machine, a program gets "stuck" whenever 'OR' is used in a listing after a 'GET' statement. For example:

```
100 PRINT "PRESS A OR B"
110 GET A$: IF A$<>"A" OR
A$<>"B" THEN 110
120 PRINT "YOU PRESSED
"A$"
130 END
140 REM
READY.
```

However, the listing works if entered as follows:

```
100 PRINT "PRESS A"
110 GET A$: IF A$<>"A"
THEN 110
120 PRINT "YOU PRESSED
"A$"
130 END
140 REM
READY.
```

To get the same result desired as shown in the first listing, I have to write it more or less this way:

```
10 GET A$
20 IF A$="A" THEN GOTO 50
30 IF A$="B" THEN GOTO 50
40 GOTO 10
50 PRINT "YOU PRESSED" A$
60 END
```

J. Rowland
Houston, Texas

A. *It seems your logic is faulty. In your first example, if you press "A" then A$ will not be equal to "B". Line 110 says that if A$ is not equal to "A", or if A$ is not equal to "B", then go back and get a new letter. We just stated A$ would not be equal to "B" so that means we must go back and get a new letter. THERE IS NO EXIT FROM THIS LINE. However, there are several ways to correctly implement this text. In addition to your final example, both of the following variations will work correctly:*

```
110 GET A$: IF A$ <> "A"
  AND A$ <> "B" THEN 110
110 GET A$: IF A$ = "A" OR
```


Continued on page 4.

Commodore — The Microcomputer Magazine *is published six times each year by Commodore Business Machines, Inc., 487 Devon Park Drive, Wayne, PA 19087. Phone: (215) 687-9750. Domestic Subscriptions: six issues, $15.00. Application To Mail At Second-Class Postage Rates Is Pending At Wayne, PA 19087 and at additional mailing offices. POSTMASTER: Send address changes to* **Commodore — The Microcomputer Magazine,** *487 Devon Park Drive, Wayne, PA 19087.*

IF YOU'RE WAITING FOR THE PRICE OF WORD PROCESSORS TO FALL WITHIN REASON,

Everyone expected it would happen sooner or later. . . with WordPro PLUS™ it already has! Now all the marvelous benefits of expensive and advanced word processing systems are available on Commodore computers, America's largest selling computer line. WordPro PLUS, when combined with the new 80 column CBM 8032, creates a word processing system comparable to virtually any other top quality word processor available—but at savings of thousands of dollars!

New, low cost computer technology is now available at a fraction of what you would expect to pay. This technology allowed Commodore to introduce the new and revolutionary CBM 8032 Computer.

WordPro PLUS turns this new CBM 8032 Computer into a sophisticated, time saving word processing tool. With WordPro PLUS, documents are displayed on the computer's screen. Editing and last minute revisions are simple and easy. No more lengthy re-typing sessions. Letters and documents are easily re-called from memory storage for editing or printing with final drafts printed perfectly at over five hundred words per minute!

Our nationwide team of professional dealers will show you how your office will benefit by using WordPro PLUS. At a price far less than you realize.

Invest in your office's future. . .
Invest in WordPro PLUS. . .
Call us today for the name of the WordPro PLUS dealer nearest you.

Professional Software Inc.
166 Crescent Road
Needham, MA 02194
(617) 444-5224
TELEX: 95 1579

Professional Software, Inc. WordPro was designed and written by Steve Punter.

Q&A HOTLINE

A$ = "B" THEN 120
115 TO TO 110

Q. I have a Hayes modem connected to the RS-232 user port of my CBM 8032. Is there a program that will enable me to use the RS-232 port and make my machine a terminal.

M. Oxman
Bethesda, MD

A. *There is a product called McTerm from Madison Computer [(208) 255-5552] which turns the user port into an RS-232 port. McTerm is a menu-driven program which allows the user to easily set and change the various communications parameters. No additional hardware is required.*

Q. I am interested in interfacing my PET 4000 series computer with a television. Are there any devices available to do this?

L. Smith
Inwood, NV

A. *Madison Computer may also solve this problem as well. They sell a circuit which converts the signals from the PET/CBM into those required by a standard video monitor.* Commodore Magazine *also intends to publish plans in a future issue for a low cost-circuit which will work with a 40 or 80 column PET/CBM.*

Q. I have a 2001-16K CBM. At times, when I am editing a program, my machine displays the following patterns:

pc	1rq	sr	ac	xr	yr sp
.;e4f2	e62e	30	4c	03	00 ed

I cannot get out of this mode without turning the machine off and losing what I have. Is there another solution to this problem?

A. *When you get the display you described, your computer has entered the machine language monitor. To exit back to BASIC, simply type X and RETURN.*

Editor's Notes
Reaffirming a Commitment

Not since Mission Control successfully concluded the last space shuttle mission had there been so much excitement in Texas. The source of all this hoopla was Commodore's unveiling of a new generation of advanced microcomputer systems at the NCC in Houston. Once again, Commodore has eagerly assumed the role of trendsetter by adding "B" series and "P" series computers to its ever-expanding range of products. Powerful yet simple to use, these innovative machines will be the talk of the industry for quite some time.

Just the sheer appearance of the new products is reason enough to get excited. Matching the beauty of its internal capabilities, the "human engineered" "B" series features attractive and fuctional housing. Its integrated green phosphor screen, which offers a very crisp display, may be adjusted using both tilt and swivel controls. And that's just the beginning!

The detachable keyboard is designed to be placed against the main unit to conserve space. However, for typing comfort, the keyboard may be placed some distance away from the main unit. Can't you imagine Commodore computerists leaning back in comfortable chairs with keyboards nestled securely in their laps.

Both "P" and "B" series micros share the same keyboard design, which reflects advanced features and comfort. Maybe you can't judge a book by its cover, but the external features of these new products are a positive indication of the computing power that's just waiting to be used.

For a thorough breakdown of these new products, be sure to read this issue's Commodore News section. And, for more information on our commitment to these products, please read on.

Obviously, when these new advanced systems become available later this year, Commodore Magazine will offer valuable information to readers. However, the reader nucleus of this magazine consists of PET, CBM and VIC users. We will not lose sight of this audience.

This publication, like the microcomputers it supports, is part of a product evolution. It will continue to reflect the substantial growth of the Commodore product line. And it will remain committed to existing Commodore products, as well as new products. Remember, this is not a PET magazine . . . or a VIC magazine . . . or "B" and "P" series publication. It is "The Microcomputer Magazine" for *all* Commodore products.

Paul Fleming

Paul Fleming
Editor

Electronic Communications

From Where I Sit

Larry Ercolino
Manager of Communications Products

When I was asked to share some views "from where I sit," I did not realize the herculean task I had undertaken. For there is so much that you need to know about my area of expertise— "Electronic Communications". It is a broad subject meant for use by computer novices, businessmen, investors, professional programmers, and students . . . In essence, anyone with a need for information or computer power not available at the local level has a genuine need in electronic communications. It is not arrogance that makes me say this is truly an area for everyone. In fact, I am certain that once your Commodore computer is linked to other computers via a phone line, you can find someone with the same interests as you. Let me explain.

The concept of "telecomputing" is not new. Universities, government agencies and businesses have been exchanging computer information by phone for many years, but until the advent of low cost modems, the equipment required was too expensive for use in small business and home computing. Recently, companies such as Compuserve, Dow Jones News/Retrieval and The Source have made access to main frame computing affordable AND practical. Suddenly, you can get high class information on your stock portfolio, even if it consists of only five shares of stock; research power is at your fingertips; comparative shopping to make the most of your buying power can be done from your living room; an electronic "post office" is open for business anytime you want—all those neat things you read about in science fiction books are really here for you to use. To bring tomorrow into your home today all you need is a Commodore computer, a modem and a telephone.

Commodore has made a major commitment to you and telecomputing. Available June 1, 1982 we will have our own information network. The Commodore Information Network will offer you a Hotline that you can access in those late hours when no one is around to help you, and the next night there will be an answer in your "electronic mailbox"! This special network will initially be accessible by you on the CompuServe Information Services network. Subscribers will be able to get current product information and technical assistance, or exchange tips with other Commodore owners. Here are a few of the services provided through the Commodore Information Network:

- Commodore Hotline
- Tech Tips
- Commodore Bulletin Board
- Product Announcements
- Software Tips
- Listing of Users' Groups
- Commodore Reference Materials
- Newsletters

Most of these items will be available by June. We will continue to increase the offerings as the system grows. Already planned is free public domain software and a program of the month. All Commodore computers will be supported in all these activities PLUS we plan to structure an ONLINE national users' group.

From the VIC-20 to the newly announced "B" and "P" series products, Commodore provides you with a cost effective and powerful entry into the growing world of telephone/computer services!

Powerful-Versatile COMMODORE-64 Computer Ideal for Home and Business Applications

The Commodore-64, one of Commodore's newest breakthroughs in adding versatility and power to home and business microcomputing, was unveiled at the Summer Consumer Electronics Show in Chicago and The National Computer Conference in Houston.

The Commodore-64, which Shearson/American Express said "could be the microcomputer industry's outstanding new product introduction since the birth of the industry," gives home and business users powerful 64K computing capability along with exciting three-dimensional-style color game graphics and music synthesis. At the suggested retail price of $595.00, the Commodore-64 offers one-third more computing power than the Apple II+® at less than one-half its suggested retail price.

Physically resembling Commodore's popular VIC 20 computer, the Commodore-64 can use VIC 20 peripherals, and goes far beyond the VIC's capabilities. With the addition of a cassette interface, the Commodore-64 can use many programs and files created for Commodore's sophisticated line of PET and CBM education- and business-oriented computers. And with the addition of an IEEE-48 cartridge, the Commodore-64 can run any Commodore peripheral, including a dual disk drive and a CBM printer.

For those programs that need conversion, the Commodore-64 has a PET emulator that makes it operate like a PET in most areas. One of its most impressive features is its ability to use a Z80 cartridge to run CP/M®, giving users access to one of the largest collections of microcomputer software available.

Computerists can use the Commodore-64's versatility to synthesize music and play all the exciting games designed for the new Commodore MAX Machine. Its 40 column x 25 line screen and 16 colors give it great flexibility for users to create their own graphics characters and games. And its 64K Random Access memory (RAM) gives it the capability for sophisticated home and business applications. Word processing programs will be available for the Commodore-64, along with Commodore's own new "electronic spreadsheet". Many new applications will be on the market soon.

The Commodore-64 relies on state-of-the-art integrated circuits designed and produced by Commodore's subsidiary, MOS Technology. The heart of the Commodore-64 system is a new microprocessor, the 6510. The 6510 is similar to the MOS 6502—the chip that made microcomputing a household occupation—but contains additional input/output (I/O) lines to handle the processing required by the new system.

Apple II+ is a registered trademark of Apple Computer, Inc.
CP/M is a registered trademark of Digital Research, Inc.

COMMODORE MAX MACHINE, 3 in 1 Computer-Game Machine-Music Synthesizer, Sure to Change the Old-Fashioned Video-Game Buying Habits of Consumers

The Commodore MAX Machine, a revolutionary three-in-one home computer-game machine-music synthesizer, was unveiled at the Summer Consumer Electronics Show (CES) at Chicago's McCormick Place.

The MAX Machine, shown in prototype at the Winter CES under the code name ULTIMAX, drew raves for its extraordinary versatility, price/performance ratio and three-dimensional-style color game graphics. Now encased in an innovative futuristic housing, it will be sold late this year with an array of arcade games, as well as educational and musical programs. At a suggested retail price of $179.95—about the same as an ordinary game machine—the Commodore MAX Machine is certain to change consumers' old-fashioned video-game buying habits.

Relying on a new display chip designed by Commodore's subsidiary, MOS Technology, the MAX Machine produces color and graphics formerly available only with a highly sophisticated character generator. And because the MAX is a real computer, users do not have to rely only on pre-programmed games, but can actually create their own games, then save them on cassette tape for future use. Its 40 column x 25 line screen and 16 colors give the MAX great flexibility for unique and exciting graphics.

Using a new Sound Interface Device, the Commodore MAX Machine produces music and sound effects that rival many of the best music synthesizers now available. The MAX produces three independent voices, each with a nine-octave range, contains a programmable ADSR (attack, decay, sustain, release) generator and a programmable filter, and has variable resonance. With these sophisticated features, the MAX Machine can command astounding orchestration when it is used with a good quality audio system.

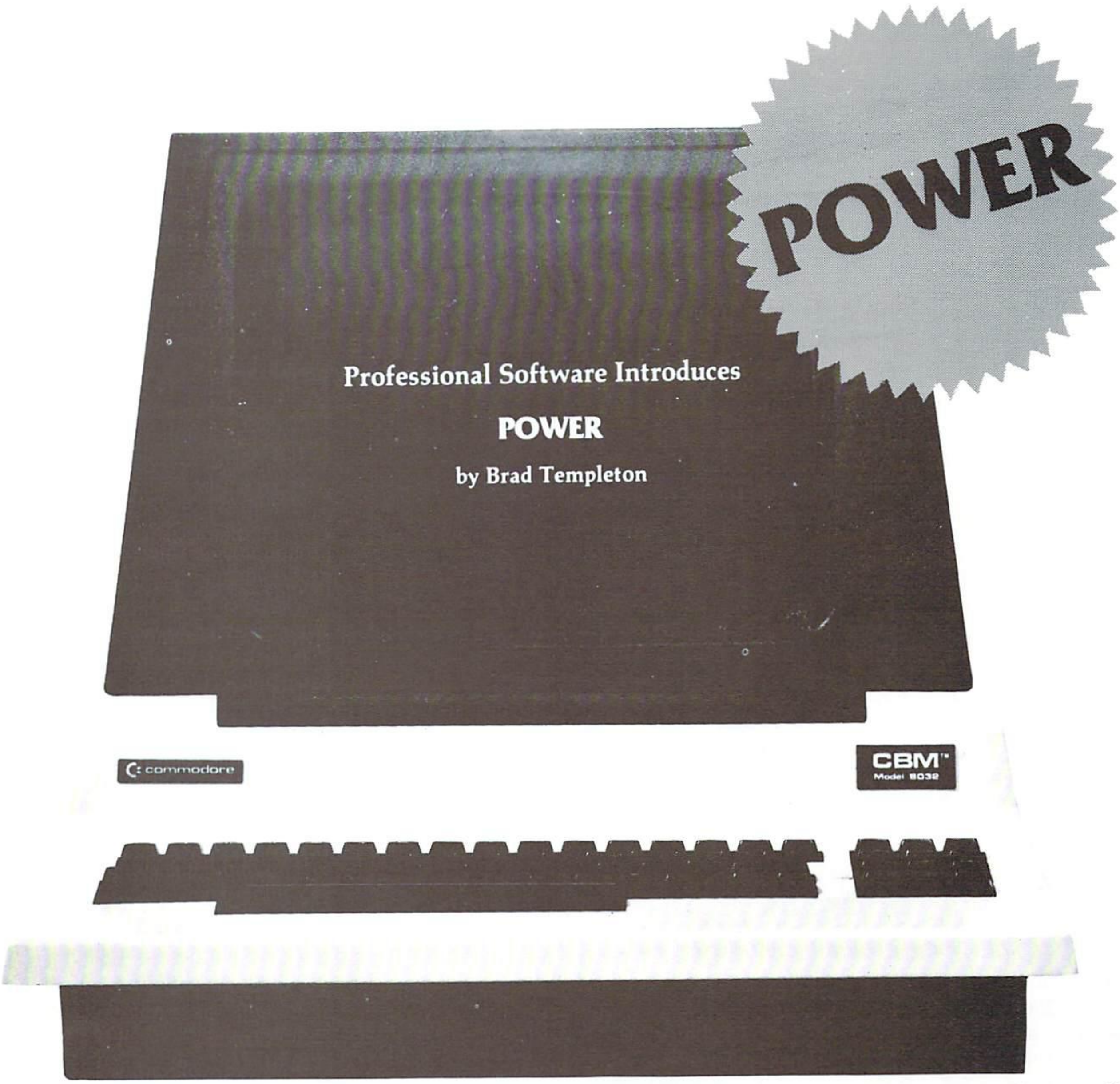
With a BASIC language cartridge, Commodore MAX Machine users can learn the fundamental language of computing and write their own programs using simple BASIC commands. MAX Machine BASIC can be translated for use with all other Commodore systems, and is capable of handling everything from word strings to math functions. With the MAX Machine's nine-digit numeric accuracy and built-in math functions, users can write a variety of useful programs for home applications.

The heart of the MAX Machine system is a new microprocessor, the 6510, designed by Commodore's MOS subsidiary.

to the 6502 chip, also designed by MOS, that made microcomputing a household activity. However, the 6510 contains additional input/outpur (I/O) lines to handle the processing required by the new system. C=

COMMODORE Quick Reference Fact Sheet

	MAX Machine	Commodore-64
Memory	2K built-in plus memory Cartridge as per BASIC specifications	64K built-in 16K ROM add-on
Screen Size	40 col x 25 lines	40 col x 25 lines
Graphics	320 x 200 pixels 255 foreground/ background color combo 16 text colors 8 movable Sprites	320 x 200 pixels 255 foreground/ background color combo 16 text colors 64 graphic characters 256 movable Sprites Independent magnification Extended Hi-res modes
Sound	6581 Sound Interface Device (SID) 3 independent voices, 9 octaves each Programmable ADSR Programmable filter Variable resonance & master vol. control	6581 Sound Interface Device (SID) 3 independent voices, 9 octaves each Programmable ADSR Programmable filter Variable resonance & master vol. control
Peripherals	Datassette Joystick Double paddle Lightpen	Datassette Joystick Double paddle Lightpen Z80 microprocessor on cartridge VIC MODEM VIC disk drive VIC graphic printer With IEEE-488 cartridge: All Commodore peripherals
Compatability	Same game cartridges will work on Commodore-64	Other BASIC Commodore programs easily converted. PET emulator to be available.



ADD POWER TO YOUR COMMODORE COMPUTER

$89.95

POWER produces a dramatic improvement in the ease of editing BASIC on Commodore's computers. POWER is a programmer's utility package (in a 4K ROM) that contains a series of new commands and utilities which are added to the Screen Editor and the BASIC Interpreter. Designed for the CBM BASIC user, POWER contains special editing, programming, and software debugging tools not found in any other microcomputer BASIC. POWER is easy to use and is sold complete with a full operator's manual written by Jim Butterfield.

POWER's special keyboard 'instant action' features and additional commands make up for, and go beyond the limitations of CBM BASIC. The added features include auto line numbering, tracing, single stepping through programs, line renumbering, and definition of keys as BASIC keywords. POWER even includes new "stick-on" keycap labels. The cursor movement keys are enhanced by the addition of auto-repeat and text searching functions are added to help ease program modification. Cursor UP and cursor DOWN produce **previous** and next lines of source code. COMPLETE BASIC program listings in memory can be displayed on the screen and scrolled in either direction. POWER is a must for every serious CBM user.

Call us today, for the name of the Professional Software dealer nearest you.

Professional Software Inc.
166 Crescent Road
Needham, MA 02194
Tel: (617) 444-5224 Telex #951579

™POWER is a registered trademark of Professional Software, Inc.

BX256, 16-BIT, 256K Multiprocessor, Tops List of New Commodore Microcomputers Unveiled at NCC

Commodore unveiled the BX256, its first advanced 16-bit multiprocessor professional microcomputer, at the National Computer Conference (NCC) in Houston (June 7-10).

Featuring 256K of user memory (RAM), the newly designed BX256 is an enhanced version of Commodore's "B" series microcomputers, and will offer two processors, including a 16-bit 8088 for CP/M-86, Release 1* compatibility. The BX256 also features an attached 80-column green phosphor screen and built-in dual disk drives. It is expandable to a maximum of 256K RAM internally with potential for 640K externally, and can accomodate an optional Z-80 processor board.

The BX256, which made its American debut at NCC after having had its world premier at the Hanover Trade Fair in Germany, will be available later this year from authorized Commodore dealers throughout the United States at a planned retail price of $2995.

In addition to internal improvements, the BX256 offers the new external "Look" of all "B" series computers. Its screen, with an 80 columns by 25 lines display, may be adjustable with separate tilt and swivel controls, and its detachable keyboard is designed to be placed against the main unit to conserve space, but may be moved away from the screen for typing comfort.

A full-sized, typewriter keyboard with a separate professional numeric keypad is standard with all 'B' series microcomputers. The keypad also includes a double zero key, a "clear entry" key and a double-size "enter key" for ease of operation. In addition to the standard individual cursor control, the B256 features 10 programmable "function keys" that can be used to perform specialized tasks.

A Commodore-designed chip—the 6581 microprocessor—gives the BX256 a full three-voice music synthesizer, with nine octave ranges. The circuit's sophistication is capable of producing many different effects. It's system includes output for direct connection to external sound systems and features a cartridge slot for instant plug-in of application and game software.

The full range of Commodore CBM peripherals are supported by the BX256 via a built-in IEEE-488 interface, and the built-in RS-232 interface allows the connection of standard computer devices such as modems and printers.

A wide variety of software will be offered for the B128 to provide users with various business, personal, entertainment, and educational applications.

The BX256 was unveiled at NCC along with two other newly announced advanced systems—the Commodore "B" and "P" series micros. In addition, two recently announced products, the Commodore-64 and Commodore "MAX" Machine, were formally introduced.

**CP/M is a registered trademark of Digital Research.*

"B Series" Product Comparison

FEATURE	"BX256"	"APPLE III"	"IBM PC"
Price[1]	$2995	$3,495	$3,045
Built-in Memory Storage	256K	128K	64K
Maximum Memory	256K internal/ 640K external	256K	256K
Disks Included (5¼″)	2	1	2
Disk Storage Included	340K	140K	320K
BASIC Language in ROM	YES	NO	YES
CP/M Software Option[2]	YES	YES	YES
Detachable Keyboard	YES	NO	YES
Real Time Clock	YES	YES	YES
Audio			
Sound Generator	YES	YES	YES
Music Synthesizer	YES	NO	NO
Video			
Display	80 Column	80 Column	80 Column
Built-in Display	YES	EXTRA	EXTRA
Input/Output			
IEEE-488 Bus	YES	NO	NO
RS-232 Communications	YES	YES	YES

[1] *Prices based on standard Apple III with 128K; IBM PC model 824 with 64K (with 256K, the IBM would cost $4,665).*
[2] *CP/M is a product of Digital Research.*

Preliminary release information. Specifications subject to change.

Commodore Announces P128 Microcomputer, 3rd Generation of its Popular PET Series

128K of RAM, Color, CP/M Compatibility Featured in Commodore's New 'P' Series

The P128 microcomputer, first of the "P" series—the third generation of Commodore's popular PET series — was one of five new advanced and entry-level micros demonstrated at the 1982 National Computer Conference (NCC) at Houston's Astrohall, June 7-10.

The P128, which connects directly to a television set or monitor via built-in RF modulator, will feature 128K of RAM, 40 columns by 25 lines display, and 16 colors for either text or graphics. The P128 will also feature a high resolution graphic display of 320 by 200 pixels.

The P128 will be available late this year from authorized Commodore dealers throughout the nation at a planned retail price of $995.00.

In addition to the basic 128K of RAM, storage in the P128 is expandable to a maximum of 256K RAM interally and 640K externally. This third-generation machine also becomes a multiprocessor system. A Commodore-designed Z-80 processor board that offers CP/M* compatibility.

Also featured in the revolutionary P128 are 10 programmable "function" keys, which can be used to perform any number of specialized tasks, as well as a number of editing keys including "individual cursor control."

Music synthesis is yet another standard feature of the new "P" series microcomputers. The P128 can stand alone or be attached directly to a high quality sound system, thus enabling users to enjoy the full range of sounds that the P128 is capable of producing.

The new P128 and all "P" Series micros support the full range of Commodore CBM peripherals via the built-in IEEE-488 interface, and the built-in RS-232 interface allows for easy connection of standard computer devices including modems and printers.

The P128 was unveiled at NCC along with two other newly announced advanced microcomputer systems—the Commodore "B" and "BX" series micros. In addition, two other new products, the Commodore-64 and Commodore "MAX" Machine, were formally introduced.

**CP/M is a registered trademark of Digital Research.*

FEATURE	P128	Apple II+
Price[1]	$995	$1,530
Built-in Memory Storage	128K	48K
Maximum Memory	512K	256K
BASIC Language in ROM	YES	YES
CP/M Software Option[2]	YES	YES
Audio		
Sound Generator	YES	YES
Music Synthesizer	YES	NO
Video		
Display	40 Column	40 Column
Color	YES	YES
Modulator Included	YES	NO
Input/Output		
IEEE-488 Bus	YES	NO
RS-232 Communications	YES	Not Inc.
Game Machine		
Cartridge Game Slot	YES	NO
Game Controller Ports	YES	YES

[1] *Price based on suggested retail for 48K Apple II+*

[2] *CP/M is a product of Digital Research.*

Preliminary release information. Specifications subject to change.

Computer Series Comparison Chart

	'P' SERIES	'B' SERIES	'BX' SERIES
USER MEMORY (RAM)			
Standard	128K	128K	256K
Maximum	256K internal/ 640K external	256K internal/ 640K external	256K internal/ 640K external
PERMANENT MEMORY (ROM)	40K	40K	40K
DISPLAY	40 Col. x 25 Line Color TV or Monitor 16 Colors 320 x 200 dot high resolution mode	80 Col. x 25 Line Integral Display Green Phosphor Screen	80 Col. x 25 Line Integral Display Green Phosphor Screen
KEYBOARD	Integral 94-key —10 function keys —Numeric keypad —Editing & Cursor Control keys	Detachable 94-key —10 function keys —Numeric keypad —Editing & Cursor Control keys	Detachable 94-key —10 function keys —Numeric keypad —Editing & Cursor Control keys
PROCESSOR (standard)	6509	6509	6509 & 8088
Optional	Z-80	Z-80	Z-80
INTERFACES	IEEE-488 RS232 CBM Cassette 8-Bit User Port Direct Audio/Video Output Cartridge Slot Control Ports	IEEE-488 RS232 CBM Cassette 8-Bit User Port Direct Audio/Output Cartridge Slot	IEEE-488 RS232 CBM Cassette 8-Bit User Port Direct Audio/Output Cartridge Slot
INTEGRAL DISKS	NO	YES	YES
Standard Language	BASIC 4.0	BASIC 4.0	BASIC 4.0
Optional Languages	CP/M*, CP/M-86 UCSD Pascal**	CP/M, CP/M-86 UCSD Pascal	CP/M, CP/M-86 UCSD Pascal
Real Time Clock	YES	YES	YES
SOUND SYNTHESIZER	YES	YES	YES

Preliminary release information. Specifications subject to change.

* *CP/M is a registered trademark of Digital Research.*
***UCSD Pascal is a registered trademark of Softech Micro Systems.*

SuperPET News from Waterloo

Waterloo Computing Systems Limited is primarily engaged in the development of portable software that will operate on a variety of computer systems. One of Waterloo's recent developments includes programming language processors designed for use with microcomputers. Their software was made available first by Commodore with the SuperPET.

6502 Assembly-Language Development System

A software package has been written for developing assembly language programs to execute on the 6502 microprocessor. Using this package, 6502 machine-language programs are prepared using the microEditor, a 6502 Assembler and a Linker which produces an executable "module" in a disk file. These components of the package operate on the SuperPET, in 6809 mode, and have functional characteristics which are almost identical to the 6809 Development System included in the original SuperPET software package (i.e., macros, structured programming, long names, bank-switch support, etc.)

Another component of the system is a program which converts a 6502 program file from Waterloo Linker format to Commodore "LOAD" format. This means that programs, which are linked to operate in the direct 32KB User RAM memory and converted, can be brought into memory for execution with the Commodore "LOAD" command. Such programs can operate on a SuperPET in 6502 mode or on other Commodore models such as the 8032 and 4032.

Two additional components of this package operate in 6502 mode on the SuperPET. Both of these components have been developed to support use of bank-switched memory for the execution of 6502 machine-language programs. The first component is a loader which reads Waterloo Linker-format files into both direct and bank-switched memory for execution. The second component is a program which supports execution of programs in bank-switched memory. Using this package, a 6502 machine-language application that makes use of the entire 96K byte RAM memory of the SuperPET can be developed conveniently.

"Bank-Switching" of Existing BASIC 4.0 Applications

An enhancement package for Commodore BASIC 4.0 facilitates use of bank-switched memory for fast operation of existing large application packages. This package operates on the SuperPET in 6502 mode.

Currently existing application packages written in BASIC 4.0 which are larger than 32K bytes, often are "modularized" into various functional component programs. Component programs are brought from disk files into memory, one at a time, for execution using the Commodore "LOAD" command. This new package makes use of the additional bank-switched memory of the SuperPET to hold copies of the component programs of an application. During execution of the application, component programs are "LOADED" from bank-switched memory at memory-to-memory transfer rates, instead of coming from disk. This facility can dramatically increase the performance characteristics of existing CBM 8032 application packages executing on the SuperPET.

Terminal Emulation Support

The existing software on the Commodore SuperPET enables it to be used as a simple terminal in passthrough mode. This support has certain shortcomings for serious use. A new software package provides substantially enhanced terminal emulation support for the Super PET. The package includes interrupt-driven, buffered input from the host line. This prevents loss of received characters during certain screen operations such as scrolling, when using data rates of up to 9600 baud.

Additional keys have been defined to enable transmission of the "BREAK" condition and the ASCII control characters. A local/remote echo setting allows optional display of characters transmitted to the host computer. Special character sequences transmitted from the host computer can be used for terminal control operations such as x-y cursor positioning, screen-clear, et cetera. Optional XON/XOFF support for both transmission and reception of data avoids data overruns without requiring large buffers. In addition to supporting the ASCII character set, an alternate mode provides APL characters with standard overstrikes.

A BYTE OF WHIMSEY...

Actual Poster Size 29" wide x 24" deep

A funny thing happened here a few months ago...this fellow walked in with this very unsophisticated art he wanted us to use on the April/May cover of the COMMODORE Magazine. Gee, how could we tell him that it wasn't the "image" we wanted to project...after all, our computers might be *FUN*, but we have some very professional people using these things — doctors, lawyers, even indian chiefs. The powers that be took one look at it and said "Hey, you can't put a lot of pixies on our cover and expect people to take you seriously... there's a pixie playing frisbee with one of our disks, and another pixie floating down a waterfall with one of our business machines, there's a funny looking toad fooling around with a VIC, there's even some pixie who thinks he can access a data base from his treehouse... our readers want to see *PEOPLE* — people looking very serious, very productive, learning all sorts of great things, not a bunch of silly pixies. Well, we took a closer look, and ran it anyway...In fact, we even had it printed as a big four-color poster to hang in your *VERY* serious office, institution of learning, or, your own treehouse... So if you have just a BYTE OF WHIMSEY in you, fill out the attached coupon and send us your check or money order.

Return to:
COMMODORE MAGAZINE
COMMODORE BUSINESS MACHINES INC.
The Meadows, 487 Devon Park Drive
Wayne, Pa. 19087

Name ____________________

Address ____________________

City ____________________

State ____________ Zip ________

Please enclose a check or money order for $5.00 for each poster ordered.

QUANTITY ____ TOTAL ENCLOSED ____

Dealer Inquiries Welcome

commodore
COMPUTER

Commodore's New B128 Microcomputer Will Offer 128K of RAM, CP/M Compatibility

Business-Oriented 'B' Series Micros Includes Built-In Screen and Dual Disk Drives

Commodore unveiled its B128 microcomputer, the second generation of its CBM professional series and flagship of its new "B" series business-oriented micros, at the National Computer Conference (NCC) at Houston's Astrohall, June 7-10.

Scheduled for release later this year, the B128 features 128K of user memory (RAM), an attached 80-column green phosphor screen and built-in dual disks. This new 'B' series computer is expandable to a maximum of 256K RAM internally with potential for 640K externally, and can accomodate an optional Z-80 processor board to provide CP/M* compatibility.

The B128 will be available later this year from authorized Commodore dealers throughout the United States at the suggested retail price of $1695.

Featuring an innovative and attractive external design, the second generation "B" series and the B128 feature an 80 columns by 25 lines display on a screen that may be adjusted to provide users with comfortable viewing angles through tilt and swivel controls.

The B128's detachable full-size keyboard is designed to be placed either next to the main unit or further away to suit the user. The keyboard has a standard separate professional numeric keypad which includes double zero and "clear entry" keys and a double size "enter key" for ease of operation. In addition to the standard individual cursor control, the B128 also features 10 programmable "function keys" that can be used to perform specialized tasks.

A Commodore-designed chip gives the B128 a full three-voice music synthesizer with nine octave ranges. The sophistication of the circuit allows the user to produce many different musical and sound effects. The new system also includes output ports for direct connection to external sound systems, and features a cartridge slot for instant plug-in of application and game software.

The full range of Commodore CBM peripherals are supported by the B128 via a built-in IEEE-488 interface, and the built-in RS-232 interface allows for the connection of standard computer devices including modems and printers.

A wide variety of software will be offered for the B128 to provide users with various business, personal, entertainment, and educational applications.

The B128 was unveiled at NCC along with two other newly announced advanced microcomputer systems—the Commodore "P" and "BX" series. Two additional new entries to Commodore's broad product line, the Commodore-64 and Commodore "MAX" Machine, were also formally introduced at the NCC.

Count on *Commodore Magazine* to bring you timely updates on these products as they become available.

**CP/M is a registered trademark of Digital Research.*

13 New Acrade-Style Games for VIC 20 Includes Bally/Midway and Scott Adams Adventure Series

Gorf, Omega Race, Wizard of Wor Now Available to Home Users Along with Wide Range of 'User Friendly' VIC 20 Peripherals

Thirteen exciting new cartridge games—including three Bally/Midway arcade games—for the phenomenally popular VIC 20 home computer were featured by Commodore at the Summer Consumer Electronics Show (CES) (June 6-9).

Also featured at the Commodore booth was a complete range of peripheral products for the VIC 20, including a single disk drive, graphic printer, VICMODEM and 16K memory expansion cartridge. These user-friendly low-cost peripherals give VIC home users the capabilities that used to be available only on complex and costly computer systems. With these peripherals and appropriate software, users can do word processing, learn programming, track finances, create their own games, or tune into telecommunications networks—to name a few possibilities.

The Bally/Midway games being shown on cartridge—Gorf, Omega Race and Wizard of Wor—are real arcade games. High resolution graphics, sound effects and play action are exactly like the popular Bally/Midway coin-operated games. Commodore's own Space Vulture, Raid on Fort Knox and Pinball Spectacular, also on view for the first time at CES, have the same kind of fast action and exciting graphics as the arcade games. Mole Attack and The Sky is Falling, Commodore's new children's cartridge games, engage tots with faster and faster action.

Commodore also introduced five challenging Scott Adams adventure games at CES—Adventureland Adventure, Pirate Cove Adventure, Mission Impossible Adventure, The Count Adventure and Voodoo Castle Adventure. These games, previously available only for high-priced systems, are cumulative fantasy games that can be saved and played over a long period of time.

Since the VIC 20 was introduced in 1981, its capabilities have continued to grow at a tremendous rate. VIC users can now learn the fundamentals of computer programming at home using Commodore's *''Introduction to BASIC''* manuals and tapes. More advanced programmers have great flexibility for desgning programs and creating graphics and sound effects, with the use of the Super Expander cartridge, Programmable Character Set Editor and VICMON machine language cartridge. Now that the VICMODEM cartridge can transform the VIC into a telecommunications terminal, businessmen, students, farmers and hobbyists have access to all the capabilities of a huge mainframe computer right in their own livingroom.

Fuller reviews of these products will follow as they become available. C=

USER CLUBS
Sound Off!

We're continuing to compile a list of all Commodore Users clubs throughout the country. If you'd like to add your name to the rolls, please send your club's name, address, and other pertinent information to:

Commodore Users Clubs
c/o Editor
Commodore Magazine
681 Moore Road
King of Prussia, PA 19406

And remember, once our list is comprehensive enough, we will begin forwarding valuable information to clubs on a regular basis, including hardware and software updates, technical bulletins, new product announcements, and troubleshooting tips.

ALABAMA

Huntsville PET Users Club
9002 Berclair Road
Huntsville, AL 35802
Contact: Hal Carey
Meetings: every 2nd Thursday

ARIZONA

VIC Users Group
1206 N. Fraser Drive
Mesa, AZ 85203
Contact: Paul V. Muffuletto

ARKANSAS

Commodore/PET Users Club
Conway Middle School
Davis Street
Conway, AR 72032
Contact: Geneva Bowlin

CALIFORNIA

Lawrence Hall of Science
UC Berkeley
Computer Project, Room 254
Berkeley, CA 94720
(415) 642-3598

PALS (PETS Around Livermore Society)
886 South K
Livermore, CA 94550
Contact: John Rambo

SCPUG Southern California PET Users Group
c/o Data Equipment Supply Corp.
8315 Firestone Blvd.
Downey, CA 90241
(213) 923-9361
Meetings: First Tuesday of each month

California VIC Users Group "VIC-VILLE"
c/o Data Equipment Supply Corp.
8315 Firestone Blvd.
Downey, CA 90241
(213) 923-9361
Meetings: Second Tues. of each month

Sacramento PET Workshop
Box 543
Davis, CA 95616
Contact: John Bowles

Commodore Users Club
1041 Foxenwoods Drive
Santa Maria, CA 93455
(805) 937-4106
Contact: Greg Johnson

Valley Computer Club
2006 Magnolia Blvd.
Burbank, CA
(213) 849-4094
1st Wed. 6 p.m.

Valley Computer Club
1913 Booth Road
Ceres, CA 95307

PUG of Silicon Valley
22355 Rancho Ventura Road
Cupertino, CA 95014

BAMBUG
1450 53rd Street
Emeryville, CA
(415) 523-7396

North Orange County Computer Club
3030 Topaz, Apt. A
Fullerton, CA 92361
Dave Smith

Lincoln Computer Club
750 E. Yosemite
Manteca, CA 95336
John Fung, Advisor

PET on the Air
525 Crestlake Drive
San Francisco, CA 94132
Max J. Babin, secretary

PALS (PETs Around Livermore Society)
886 South K
Livermore, CA 94550
(415) 449-1084
Every third Wednesday 7:30 p.m.
Contact: J. Johnson

SPHINX
314 10th Avenue
Oakland, CA
(415) 451-6364
Every 2nd & 4th Thurs.

San Diego PUG
c/o D. Costarakis
3562 Union Street
(714) 235-7626
7 a.m.-4 p.m.

Walnut Creek PET Users Club
1815 Ygnacio Valley Road
Walnut Creek, CA 94596

Jurupa Wizards
4526 Kingsbury Pl.
Riverside, CA 92503
Contact: Walter J. Scott

CONNECTICUT

John F. Garbarino
Skiff Lane Masons Island
Mystic, CT 06355
(203) 536-9789

Commodore User Club
Wethersfield High School
411 Wolcott Hill Road
Wethersfield, CT 06109
Contact: Daniel G. Spaneas

VIC Users Club
c/o Edward Barszczewski
22 Tunxis Road
West Hartford, CT 06107

FLORIDA

Jacksonville Area PET Society
401 Monument Road, #177
Jacksonville, FL 32211

Richard Prestien
6278 SW 14th Street
Miami, FL 33144

South Florida PET Users Group
Dave Young
7170 S.W. 11th
West Hollywood, FL 33023
(305) 987-6982

VIC Users Club
c/o Ray Thigpen
4071 Edgewater Drive
Orlando, FL 32804

ILLINOIS

Shelly Wernikoff
2731 N. Milwaukee Avenue
Chicago, IL 60647

Central Illinois PET Owners
Rick Goldsmith
2730 Townway Road #E-54
Danville, IL 61832

PET VIC Club (PVC)
40 S. Lincoln
Mundelein, IL 60060
Contact: Paul Schmidt, president

Rockford Area PET User's Group
1608 Benton Street
Rockford, IL 61107

Commodore Users Club
1707 East Main St.
Olney, IL 62450
Contact: David E. Lawless

INDIANA

PET Users
Jerry Brinson
PO Box 36014
Indianapolis, IN 46236
(317) 898-3604

GHS Computer Club
c/o Grangeville High School
910 S. D St.
Grangeville, ID 83530
Contact: Don Kissinger

Cardinal Sales
6225 Coffman Road
Indianapolis, IN 46268
(317) 298-9650
Contact: Carol Wheeler

CHUG (Commodore Hardware Users Group)
12104 Meadow Lane
Oaklandon, IN 46236
Contact: Ted Powell

IOWA

PET Users Group
c/o Don Vorhies
1321 42 St. SE
Cedar Rapids, IA 52403

KANSAS

Wichita Area PET Users Group
2231 Bullinger
Wichita, Kansas 67204
(316) 838-0518
Contact: Mel Zandler

Kansas Commodore Computer Club
101 S. Burch
Olathe, KS 66061
Contact: Paul B. Howard

MARYLAND

Assoc. of Personal Computer Users
5014 Rodman Road
Bethesda, MD 20016

Blue TUSK
700 East Joppa Road
Baltimore, MD 21204
Contact: Jim Hauff

MASSACHUSETTS

Eastern Massachusetts VIC Users Group
c/o Frank Ordway
7 Flagg Road
Marlboro, MA 02173

VIC Users Group
c/o Ilene Hoffman-Sholar
193 Garden St.
Needham, MA 02192

MICHIGAN

David Liem
14361 Warwick Street
Detroit, MI 48223

PET User Group
Peter Oakes
2235 Lakeshore Drive
Muskegon, MI 49441

Toledo PETS
734 Donna Drive
Temperance, MI 48182
Contact: Gerald Carter, president

Commodore User Club
32303 Columbus Drive
Warren, MI 48093
Contact: Robert Steinbrecher

Commodore Users Group
c/o Family Computer
3947 W. 12 Mile Rd.
Berkley, MI 48072

MINNESOTA

MUPET (Minnesota Users of PET)
P.O. Box 179
Annandale, MN 55302
c/o Jon T. Minerich

MISSOURI

Clearwater Club
Clearwater School
Star Route
Piedmont, MO 63957
Contact: Carolyn Polk

St. Louis Club
Mary Perkinson
46 Westwood Court
St. Louis, MO 63131
(314) 432-5225

MONTANA

Powder River Computer Club
Powder River County High School
Broadus, MT 59317
Contact: Jim Sampson

Commodore User Club
1109 West Broadway
Butte, MT 59701
Contact: Mike McCarthy

NEVADA

Las Vegas PET Users
4884 Iron Avenue
Las Vegas, NV 89110

NEW JERSEY

Amateur Computer Group of New Jersey
John Loofbourrow
UCTI, 1776 Raritan Road
Scotch Plains, NJ 07076
(201) 233-7068

Amateur Computer Group
18 Alpine Drive
Wayne, NJ 07470

Somerset Users Club
49 Marcy Street
Somerset, NJ 08873
Contact: Robert Holzer

NEW HAMPSHIRE

Northern New England Computer Society
PO Box 69
Berlin, NH 03570

NEW YORK

Capital District PET Users
Ben Green
Albany Area, NY
(518) 370-1820

Long Island PET Society
Ralph Bressler
Harborfields HS
Taylor Avenue
Greenlawn, NY 11740

PET User Club of Westchester
Box 1280
White Plains, NY 10602
Contact: Ben Meyer

LIVE (Long Island VIC Enthusiasts)
17 Picadilly Road
Great Neck, NY 11023
Contact: Arnold Friedman

PET User Group
Westchester, NY
(914) 428-7872
Every 2nd Tuesday

PET User Group
c/o Meyer
35 Barker Avenue
White Plains, NY 10610

Commodore Masters
25 Croton Ave.
Staten Island, NY 10301
Contact: Stephen Farkouh

VIC Users Club
76 Radford St.
Staten Island, NY 10314
Contact: Michael Frantz

VIC Users Club
c/o Christopher Kwasnicki
44 Harvey Ave.
Staten Island, NY 10314

PUG of Rockland County
c/o Ross Garber
14 Hillside Court
Suffern, NY 10901
(914) 354-7439

NORTH CAROLINA

Amateur Radio PET User's Group
PO Box 30694
Raleigh, NC 27622
Contact: Hank Roth

Commodore Users Club
4241 Castleton Road
Charlotte, NC 28211
Contact: Ed Harris

VIC Users Club
c/o David C. Fortenberry
Route 3, Box 351
Lincolnton, NC 28092

OHIO

Dayton Area PET User Group
933 Livingston Drive
Xenia, OH 45385
B. Worby, president
(513) 848-2065
J. Watson, secretary
(513) 372-2052

Central Ohio PET Users Group
107 S. Westmoor Avenue
Columbus, Ohio 43204
(614) 274-6451
Contact: Philip H. Lynch

OREGON

NW PET Users Group
John F. Jones
2134 N.E. 45th Avenue
Portland, OR 97213

PENNSYLVANIA

PET User Group
Gene Beals
PO Box 371
Montgomeryville, PA 18936

Penn Conference Computer Club
c/o Penn Conference of SDA
720 Museum Road
Reading, PA 19611
Contact: Dan R. Knepp

PACS PET Users Group
20th & Olney Streets
Philadelphia, PA

Glen Schwartz
807 Avon
Philadelphia, PA 19116

Gene Planchak
4820 Anne Lane
Sharpsville, PA 15150
(412) 962-9682

PPG (Pittsburgh PET Group)
c/o Joel A. Casar, DMD
2015 Garrick Drive
Pittsburgh, PA 15235
(412) 371-2882

SOUTH DAKOTA

PET User Group
515 South Duff
Mitchell, SD 57301
(605) 996-8277
Contact: Jim Dallas

TENNESSEE

River City Computer Hobbyists
Memphis, TN
1st Mon. at Main Library

TEXAS

SCOPE
1020 Summit Circle
Carrolton, TX 75006

PET Users
2001 Bryan Tower
Suite 3800
Dallas, TX 75201

Larry Williams
PO Box 652
San Antonio, TX 78293

PET User Group
John Bowen
Texas A & M Microcomputer Club
Texas A & M, TX

UTAH

Utah PUG
Jack Fleck
2236 Washington Blvd.
Ogden, UT 84401

The Commodore User's Club
742 Taylor Avenue
Ogden, Utah 84404
Contact: Todd Woods Kap, president;
David J. Shreeve, vice president

The VIClic
799 Ponderosa Drive
Sandy, UT 84070
Contact: Steve Graham

VIRGINIA

Northern VA PET Users
Bob Karpen
2045 Eakins Court
Reston, VA 22091
(803) 860-9116

VIC Users Group
Rt. 2, Box 180
Lynchburg, VA 24501
Contact: Dick Rossignol

VIC Users Group
c/o Donnie L. Thompson
1502 Harvard Rd.
Richmond, VA 23226

WASHINGTON

NW PET Users Group
2565 Dexter N. 3203
Seattle, WA 98109
Contact: Richard Ball

PET Users Group
c/o Kenneth Tong
1800 Taylor Ave. N102
Seattle, WA 98102

WISCONSIN

Sewpus
c/o Theodore J. Polozynski
PO Box 21851
Milwaukee, WI 53221

CANADA

Toronto PET Users Group
381 Lawrence Ave. West
Toronto, Ontario, Canada
M5M 1B9
(416) 782-9252
Contact: Chris Bennett

PET Users Club
c/o Daniel Cayer
R.R. 6
Simcoe, Ontario
Canada N3Y 4K5

Vancouver PET Users Group
Box 91164
West Vancouver, British Columbia
Canada V7V 3N6

KOREA

Commodore Users Club
K.P.O. Box 1437
Seoul, Korea
Contact: S. K. Cha

MEXICO

Asociacion De Usuarios Commodore
c/o Alejandro Lopez Arechiga
Holbein 174-6° Piso
Mexico 18, D.F.

NORWAY

VIC Club of Norway
Nedre Bankegt 10,
1750 Halden
Norway

Everything You Always Wanted To Know About Commodore Computers*

*And Asked!

PET

1. Q: Can the memory of the 4000 series computers be expanded beyond 32K?

A: The memory of the 4000 series computers can be expanded provided the machine has a universal board and 32K RAM. The 64K expansion board will provide 96K of RAM.

2. Q: Where can a conversion kit for the keyboard of the original PET 2001 computer be found?

A: The original flat keyboard of the 2001 series can be converted to a regular typewriter keypad by using keyboard interfaces manufactured by Skyles Electric Works of Mountain View, California, or by George Risk Industries of Kimball, Nevada.

3. Q: How can I tell which set of ROMs my PET 2001 computer has?

A: Upon power-up, the BASIC message that appears on the screen will indicate which set of ROMs the PET 2001 has. If an "*" sign appears, it is the original BASIC 1.0 machine. If a "#" sign appears, it is a BASIC 2.0 or higher machine.

4. Q: How many cassette recorders can be connected to the 4016 computer?

A: The 4016 has two cassette ports, which enables you to connect two cassette recorders at one time.

5. Q: Can WordPro 4 be used with the PET 2001 computer?

A: No, the PET 2001 is a 40-column machine and WordPro is designed to work with an 80-column machine. However, WordPro 1, 2, and 3 are all 40-column versions and can be used with the PET 2001 computer.

6. Q: Which disk drive is recommended for use with the PET 2001 computer?

A: All Commodore disk drives work equally well with the PET 2001. Customers should choose the one that fills their storage requirements and budget.

7. Q: How is the PET interfaced to a Diablo printer?

A: To interface the PET to a Diablo, use an ADA-1450 adapter. This part is included with all Commodore 8300 printers.

8. Q: With respect to the PET 2001 computer and the 2040 disk drive, how can a record be deleted?

A: To delete a record on file, create a new file (possibly with the same name) and write all of the records to it—with the exception of the one to be deleted—then scratch the original file.

CBM

1. Q: What languages are available for the 8032?

A: The 8032 is capable of executing FORTH, PASCAL, LISP, COMAL, and PILOT, as well as Microsoft BASIC and 6502 Assembler. With the 64K Memory Expansion Board, the 8032 is capable of executing UCSD PASCAL.

2. Q: What is a memory map?

A: A memory map is a list of all the specialized memory addresses in the PET/CBM/VIC computers. A separate map is required for each version of BASIC.

3. Q: Can the 8032 interface with a TV monitor?

A: The 8032 can interface with a monitor, but a high-resolution monitor (16 MHz or greater bandwidth) is required.

4. Q: What is the "spacemaker," and how is it utilized?

A: The spacemaker is a ROM-switching device. Empty sockets on the main logic board are used by various programs which require insertion of a particular ROM. Rather than removing and inserting ROMs whenever

a particular program is run, all ROMs can remain in place and the user can switch between them manually or under program control. Additional information can be obtained from Micro Tech in Langhorn, PA; (215) 757-0284.

5. Q: Can files made in 6502 mode be read in 6809 mode and vice versa?

A: Yes, with software translation, files made in 6502 mode can be read in 6809 mode and vice versa. The translation needed for 6502 mode is PET ASCII, and the translation needed for 6809 mode is U.S.ASCII.

6. Q: Where is the cursor address stored in the 8032?

A: On the 8032, the column position of the cursor is decimal address 198. The row position is decimal address 216. This information was obtained from the October issue of *Commodore Microcomputer Magazine*.

7. Q: How can three portions of one program be merged together when using the 8032 and the 8050 disk drive?

A: Programs such as the "Programmer's Toolkit" allow merging of programs. Commando Chip and its merge command can be used, or programs can be merged with machine language programming.

8. Q: How is "pi" accessed on the 8032?

A: To access "pi" on the 8032, PRINT a CHR$(255) while in the graphics mode. If you are in upper case/graphics mode, POKE 32768,94 and this will put a "pi" in the upper-left corner of the screen.

9. Q: How fast does the 8032 transfer data to the 8050 disk drive?

A: The rate of the IEEE bus is approximately 9 kilobytes per second.

10. Q: Can the character format be changed on the 8032?

A: The character set may be changed by programming your own EPROM in place of the character generator ROM.

SuperPET

1. Q: When using the modem with the SuperPET, why do characters sometimes seem to be echoed back onto the screen?

A: The built-in communications program on the SuperPET prints the characters on the screen as they are keyed in from the keyboard. If the host computer is set to echo, then the received character will be a duplicate, causing the screen to display each character twice. Almost every host computer can be made to not echo. That command depends on the host computer being used.

2. Q: Does the SuperPET have double precision?

A: Double precision implies twice the normal number of digits of accuracy. The SuperPET allows for nine digits of accuracy for all operations, which is better than single precision on other systems.

3. Q: Is there a compiler being developed for the SuperPET?

A: At this time, compilers are being developed for all the SuperPET languages written by Waterloo.

4. Q: What pins are required to connect a modem to the SuperPET?

A: Pins 2, 3, 5, 6, 7, 8, and 20, are required to connect a modem to the SuperPET. A straight-through 25 pin RS232 modem cable works very well and is inexpensive.

5. Q: How can the screen be cleared from within the program on the SuperPET?

A: The screen cannot be cleared from within the program. However, an alternative is to print 25 blank lines, which will cause the text to scroll off the screen.

6. Q: Is all of the RAM accessible for programming on the SuperPET?

A: Assembler Language programs can access all 96K of RAM in the computer. The higher level languages load the interpreter into upper 64K of RAM and leave the lower 32K of RAM for programming.

7. Q: What type of communication signal does the SuperPET send?

A: The SuperPET will receive and transmit a standard RS232 signal at 300 to 9600 baud over the built-in RS232 port. Of course, the SuperPET still has the IEEE port and the User port.

8. Q: What ASCII is used with the 6809 and the 6502 microprocessors?

A: The 6809 microprocessor utilizes standard ASCII (American Standard Code for Information Interchange), while the 6502 microprocessor uses a variation called PET ASCII.

9. Q: How much memory is available for programming in the various SuperPET languages?

A: All the SuperPET interpreters (except APL) leave approximately 30 kilobytes available for programming. APL provides a workspace of approximately 28K. Using 6809 or 6502 Assembler programs, the entire 96K is available.

10. Q: What functions will the COBOL language interpreter include?

A: The Waterloo microCOBOL is a subset of ANSI-1974 Standard COBOL. All of the Level 1 features including the NUCLEUS, SEQUENTIAL I/O, RELATIVE I/O, and TABLE-HANDLING are supported. Certain features of Level 2 are also

COMMODORE NEWS

supported including the PERFORM, STRING, UNSTRING, and INSPECT verbs.

VIC 20 Personal Computer

1. Q: Is the VICModem "switchable" from full-duplex to half-duplex?
A: The VICTerm I cassette tape that is included with the VICModem allows automatic selection of either full-duplex or half-duplex.

2. Q: How is bit mapping accomplished on the VIC 20?
A: Bit mapping is accomplished by putting consecutive characters on the screen, then re-programming the dots of the characters. This concept is elaborated in the Programmer's Reference Guide.

3. Q: How are "xy" graphics plotted on the VIC 20?
A: To plot low-resolution graphics, use the cursor control characters imbedded in PRINT statements. For high-resolution plotting, the VIC Super Expander cartridge is recommended.

4. Q: How is the clock displayed on the VIC 20?
A: To display the clock on the VIC 20, PRINT TI$ by PRINTing TI$="033000". This means 3:30 and no seconds.

5. Q: Is PASCAL available for the VIC 20?
A: PASCAL is available for the VIC 20 from Abacus Software, which is located in Grand Rapids, Michigan, (616) 241-5510.

6. Q: Is there a way to determine how much memory is left in the VIC after RUNning a program?
A: You can determine how much memory is left after RUNning a program by typing:
PRINT FRE(0)

7. Q: Do the various application software cartridges reduce the amount of RAM available?
A: The Programmer's Aid and VICMON Machine Language Monitor cartridges do not reduce the amount of RAM accessible to BASIC. The VIC 20 Super Expander adds 3 kilobytes to RAM.

8. Q: What telecommunications networks work with the VIC 20?
A: The VICModem allows the VIC to access such networks as The Source, Compuserv, Micronet, Dow Jones, and the New York Times, to name a few.

9. Q: How many digits does the VIC floating point have?
A: The floating point variable routines in the VIC have nine significant digits for the mantissa, and the exponent in the range of −38 to +37.

10. Q: What is the difference in the voltage levels of the VIC's RS232 and standard RS232?
A: The VIC voltages are at TTL level (0 to 5 volts), while the RS232 standard is −12 to +12 volts. In addition, the signal levels from the VIC are inverted from the standard RS232.

Disk Drives

1. Q: Is it possible to use hard-sectored diskettes with the 4040 disk drive?
A: Yes, it is possible to use hard-sectored diskettes with a 4040 disk drive because it has no sector-hole detect sensor. The 4040 disk drive ignores the sector holes in the diskette so any kind will work.

2. Q: How can relative files be copied from a 4040 to an 8050 disk drive?
A: To copy relative files from a 4040 to an 8050 disk drive, the side sector blocks have to be copied. This can be done with the "Copy All" program.

3. Q: Can several PET/CBM computers be connected to a disk drive?
A: Although it is not recommended, it is possible to cable together more than one computer to a disk drive. Be careful to have only one computer access the disk drive at any one moment, otherwise the system may crash. One way to prevent this problem is to use the MUPET, which is sold by Canadian Micro Distributors of Milton, Ontario; (416) 878-7277. This device allows connection of up to eight computers to one disk drive. Another way is to use the REGENT, which is sold by Skyles Electric Works

of Mountain View, California; (415) 965-1735. This device allows connection of up to sixteen computers to one drive and it works especially well in a school environment.

4. Q: Can more than one disk drive be connected to a computer?

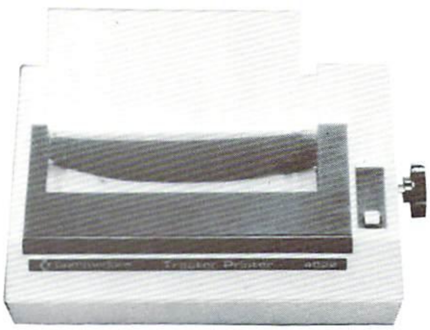
A: Up to eight disk drive units can be connected to a single computer. It is possible to set the unit address on each drive as a device number between "8" and "15". This can be done under program control or as a hardware modification.

5. Q: Where can an explanation of random, sequential, and relative files be found?

A: For a detailed explanation of these files, refer to the *PET/CBM Personal Computer Guide (Second Edition)*. This book is available from your authorized Commodore dealer, or it can be obtained directly from the publisher, Osborne/McGraw-Hill.

6. Q: Is it necessary to load the DOS into the disk drive?

A: The DOS (Disk Operating System) is included in ROM on the main logic board of the disk drive. Therefore, it is not necessary to load the DOS—the drive is ready when it is powered on.



Printers

1. Q: Do Commodore printers require unique type of paper?

A: Most Commodore printers are capable of using ordinary bond paper. Paper for all printers is available from authorized Commodore dealers.

2. Q: Can legal-size paper be used with the 8023 printer?

A: Standard 11" x 17" computer paper can be used with the 8023 printer.

3. Q: Can the 2022 printer handle mixed formatted data?

A: The 2022 printer can handle both numeric and alpha characters.

4. Q: What printers are compatible with the 8032?

A: All Commodore printers are compatible with the 8032. By using a PET to serial interface, some ASCII printers will also work.

5. Q: Where can necessary supplies for printers be obtained?

A: Most authorized Commodore dealers stock ribbons, paper, and other supplies for use with Commodore printers.

6. Q: Is the 8300 printer supplied with a tractor-feed mechanism?

A: The tractor-feed mechanism is an optional feature for the 8300 printer, and it allows bi-directional (up/down) movement of the paper. This device is available from your authorized Commodore dealer.

7. Q: Does Commodore manufacture interfaces to allow the use of non-Commodore printers?

A: There are vendors that manufacture interfaces to allow the use of some non-Commodore peripherals. However, Commodore peripherals do not require interfaces for use with Commodore computers.

8. Q: How can a program be listed in upper/lower case on the 8023 printer?

A: Set the printer to upper/lower case by issuing this sequence of commands:

```
OPEN 1,4,7
PRINT#1
CLOSE1
```

The printer will now print in upper/lower case.

9. Q: What are the misprints in the 4022 printer manual?

A: On page 27, the last line in the example should read: PRINT#8, CHR$(20).

On page 31, the second line in the first example should read:

```
PRINT#6, CHR$(195)
```

This value will produce one line per inch. The proper value to use for six lines per inch is 36.

10. Q: Can the printer be made to print a certain number of lines per page?

A: This technique is called "paging" and can be utilized on the 2022, 4022, and the 8023 printers. The command to turn this feature on is:

```
OPEN 1,4
PRINT#1, CHR$(147)
```

This command sets the number of lines per page to 66. To set a different number of lines per page, for example, 40, follow the first command with:

```
OPEN 2,4,3
PRINT#2, CHR$(40)
```

The number in the parenthesis indicates the number of lines per page desired. To turn paging off, type: PRINT#1, CHR$(19).

More on Planning for Computers in Education

Dr. Bruce Downing
Director, Industry-Marketing
Commodore

In the last issue of *Commodore Magazine* we outlined some of the most important concepts that need to be addressed in adequately planning for the use of computers in education. Key elements of a plan include setting educational objectives, designing and administrative strategy, training teachers and administrators, selecting hardware and software and deciding on the type of support personnel needed.

In this issue we will explore in more depth some of the issues related to setting objectives and administration.

Many of the readers of this magazine are already fairly adept at using computers in a variety of interesting ways. Some find it difficult to understand why the use of computers in schools is not more generally accepted, since their advantages are so obvious to the computer buff. However, the educational advantages of computers are not fully understood by the great majority of educators. Many have the attitude that the computer as an educational tool is only a passing fad.

In fairness to those doubting educators, some computer applications have been oversold or presented as easy solutions to difficult educational problems. At the same time, the widespread use of computer technology in our society creates an enormous pressure for schools to "do something about computer literacy." And this pressure is being directed at schools with decreased financial resources and teachers who have never had the opportunity to learn about computers. Add to this picture the fact that microcomputers such as the Commodore VIC-20 and the new Commodore 64 are so low priced and have so many capabilities that home applications, including education, will become very popular. Not surprisingly, many children are learning to use computers outside of school.

The need to tackle the question of computers in education has been recognized by the vast majority of schools. Just getting started has been a major problem, so perhaps setting objectives that make educational sense should be the first consideration.

Four basic areas of potential computer use can be identified for these objectives: computer literacy, computer-assisted instruction, computer-managed instruction and administrative applications.

Computer Literacy

This is one of the most viable areas to begin a computer literacy program. Since computers are a daily part of our lives, it makes sense to develop curricula which will help students from pre-school through adult continuing education programs understand the basics of computing. Although the number of topics that could be taught in such curricula is massive, the primary areas of concentration for most computer literacy programs involve basic understanding of computers, some level of competent use and special applications training.

Programs designed to develop a basic understanding usually involve topics such as:

- An introduction to keyboards.
- The functions of the parts of a computer (processor, memory, mass storage, printers, etc.)
- What is a program and programming language?
- Typical applications of computers.
- Historical and societal impact of computers.

Curriculum programs designed to teach the competent use of computers are usually started at grade six or above, although some schools have been successful teaching basic programming skills to younger children. Programs for teaching competence with computers often involve the teaching of one or more programming languages. High school programs are sometimes viewed as preparation for college level computer science curricula.

Finally, computer literacy in adult education programs is often successful when the curriculum is designed to teach computer applications such as word processing, data entry and data retrieval. Also, programs addressing small business applications in areas such as accounting and inventory control are becoming popular.

Computer-Assisted Instruction

Applications in which computers are used to deliver part of the instruction to students have been categorized as computer-assisted instruction (CAI). The use of the computer for drill and practice as a tutor, for providing simulations and as a problem solving tool are the most typical applications of CAI.

Setting meaningful objectives for the use of computers in

direct instruction is often difficult. The most effective uses in this area are viewed as supplements to other instructional techniques. In addition, effective CAI materials are designed to teach specific topics that represent learning problem areas. Drill and practice for students who need help or simulation of otherwise costly science experiments are good examples. Again, the use of the computer is based upon an identified need—then the educational objectives become fairly clear.

Computer-Managed Instruction

Computer applications such as test generation, test scoring and classroom record keeping are designed to help instructors manage the learning process. Such applications are termed computer-managed instruction (CMI). Along with computer literacy, CMI is a good candidate for the initial entry into computer applications.

Trends toward competency-based education, in which student progress is constantly measured and recorded, has created a large demand for automated procedures. Indeed, the information gathered through testing programs can point to areas of learning problems. After the problem areas are more clearly defined through CMI methods, then CAI techniques can be considered to address the problem areas. Very good software is available on microcomputers to help teachers test individual students' skills levels and track learning improvement regardless of the instructional techniques.

Administrative Applications

While educators are evaluating the applications of microcomputers for instructional support, they should also examine the possible administrative applications. Multiple applications available for microcomputers can often help justify their purchase. Obvious administrative applications include attendance recording and reporting, word processing, room scheduling, grade reporting and inventory control.

The applications I've described typically grow out of the needs of students and educators. Setting objectives for how computers can be helpful can also be difficult—particularly when various teachers and administrators will have different ideas on the priorities for multiple objectives. It is extremely important to develop a planning and administrative capability that will lead to useful and effective applications. The school administration should not make unilateral decisions on computer plans, nor should the special biases of a few who have experience in all of the potential application areas mentioned above. Therefore, setting objectives should be a process involving a wide group of interests, including the administration, teachers, the school board (or comparable group) and community members such as parents.

It is not realistic to set priorities with a large, representative committee, but input on potential objectives should be solicited from a broad spectrum of interested people. During this stage, individuals who are somewhat computer literate should resist the temptation to press for specific hardware and software, but concentrate on educational needs and objectives. Outside consultants, fact-finding committees and experienced users at other schools can be very useful at the initial interview.

Once the initial objectives are set, then hardware and software selection and training activities will become much more direct and less confusing. Exploring hardware and software alternatives early in the planning process can help clarify the possibilities but major purchases should be delayed until clear objectives are set.

In subsequent articles we will explore procedures for hardware and software selection and examine model teacher training programs. C=

Books for Use with Commodore Computers

There are a number of good books available for use with Commodore computers, most of which can be ordered through Micromedia, 61 South Lake Avenue, P.O. Box 4509, Pasadena, California 91106. A catalog is available upon request.

If you are aware of any other publications that should be on this list, please let us know.

PUBLISHER: **Hayden Books, Inc.**
Rochelle Park
New Jersey

Title: I SPEAK BASIC TO MY PET
Author: Aubrey Jones, Jr.
Price: Package of 20 student texts; teacher's edition and exam set — $156.25
Individual copies of teacher's edition — $16.20
Student texts — $7.45
Exam set — $12.50

Title: LIBRARY OF PET SUBROUTINES
Author: Nick Hampshire
Price: $14.95

Title: PET GRAPHICS
Author: Nick Hampshire
Price: $16.95

Title: BASIC CONVERSIONS HANDBOOK Apple, TRS 80, and PET
Authors: David A. Brain/Phillip R. Oviatt/Paul J. Paquin/Chandler P. Stone
Price: $7.95

PUBLISHER: **Reston Publishing Company**
(Prentice-Hall)
Reston, VA 22090

Title: PET BASIC — TRAINING YOUR PET COMPUTER
Author: Ramona Zamora/Wm. F. Carvie/ B. Allbrecht
Price: $14.95

Title: PET GAMES AND RECREATION
Author: M. Ogelsvy/L. Lindsey/D. Kunkin
Price: $12.95

Title: PET BASIC
Author: Richard Huskell
Price: $12.95 — Available June 1982

Title: VIC GAMES AND RECREATION
Price: $12.95 — Available June 1982

PUBLISHER: **P. C. Publications**
3140 Harmony Hill Road
Placerville, CA. 95667

Title: BEGINNING SELF TEACHING COMPUTER LESSONS
Price: $10.00

PUBLISHER: **McGraw-Hill**
8171 Redwood Highway
Novato, CA 94947

Title: HOME AND OFFICE USE OF VISICALC
Author: D. Castlewitz/L. Chisauki
Price: $14.95

Title: HANDS-ON BASIC WITH A PET
Author: H. Peckman
Price: $13.95

PUBLISHER: **Osborne/McGraw-Hill**
630 Bancroft Way
Berkeley, CA 94710

Title: PET FUN AND GAMES
Author: R. Jeffries/G. Fisher
Price: $10.00

Title: PET AND THE IEEE
Author: A. Osborne/C. Donahue
Price: $15.00

Title: SOME COMMON BASIC PROGRAMS FOR THE PET
Author: L. Poole/M. Borchers/C. Donahue
Price: $14.99

Title: CBM PROFESSIONAL COMPUTER GUIDE
Price: $15.00 — available June 1982

Title: THE PET PERSONAL GUIDE
Price: $15.00 — available June 1982

PUBLISHER: **Total Information Services**
P.O. Box 921
Los Alamos, CA 87544

Title: UNDERSTANDING YOUR PET/CBM VOL. 1 BASIC PROGRAMMING
Price: $14.95

Title: UNDERSTANDING YOUR VIC
Author: David Schultz
Price: $11.95

PUBLISHER: **Dilithium Press**
11000 S.W. 11th Street
Beaverton, OR 97005

EDUCATION

Author: Tom Rugg/Phil Feldman
Price: $19.95
Title: PET BASIC
Available August 1982

PUBLISHER: **Compute**
625 Fulton Street
Greensboro, NC 27403

Title: COMPUTE'S FIRST BOOK OF PET/CBM
Price: $12.95

PUBLISHER: **Telmas Courseware Ratings**
115 N. Oak Park Ave.
Oak Park, Illinois 60301 (312-848-4000)

Title: BASIC AND THE PERSONAL COMPUTER
Author: T. A. Dwyer/M. Critchfield
Price: $12.95

PUBLISHER: **Prentice-Hall, Inc.**
P. O. Box 500
Inglewood Cliffs, NJ 07632

Title: THE PET PERSONAL COMPUTER FOR BEGINNERS
Author: S. Dunn/V. Morgan
Price: $10.95

PUBLISHER: **Creative Computing**
P.O. Box 789
Morris, Illinois 07960 (800-526-0666)

Title: GETTING ACQUAINTED WITH YOUR VIC-20
Author: T. Hartnell
Price: $8.95

PUBLISHER: **Cowbay Computing**
P.O. Box 515
Manhasset, NY 11030

Title: FEED ME, I'M YOUR PET COMPUTER
Author: Carol Alexander
Price: Workbook I — $4.95

Title: LOOKING GOOD WITH YOUR PET
Author: Carol Alexander
Price: Workbook II — $4.95

Title: TEACHER'S PET — PLANS, QUIZZES AND ANSWERS
Price: $4.00

PUBLISHER: **Howard W. Sams**
4300 W. 62nd Street, P.O. Box 7092
Indianapolis, Indiana 46206
(800-428-3696)

Title: MOSTLY BASIC APPLICATIONS FOR YOUR PET
Author: Howard Berenbon
Price: $12.95

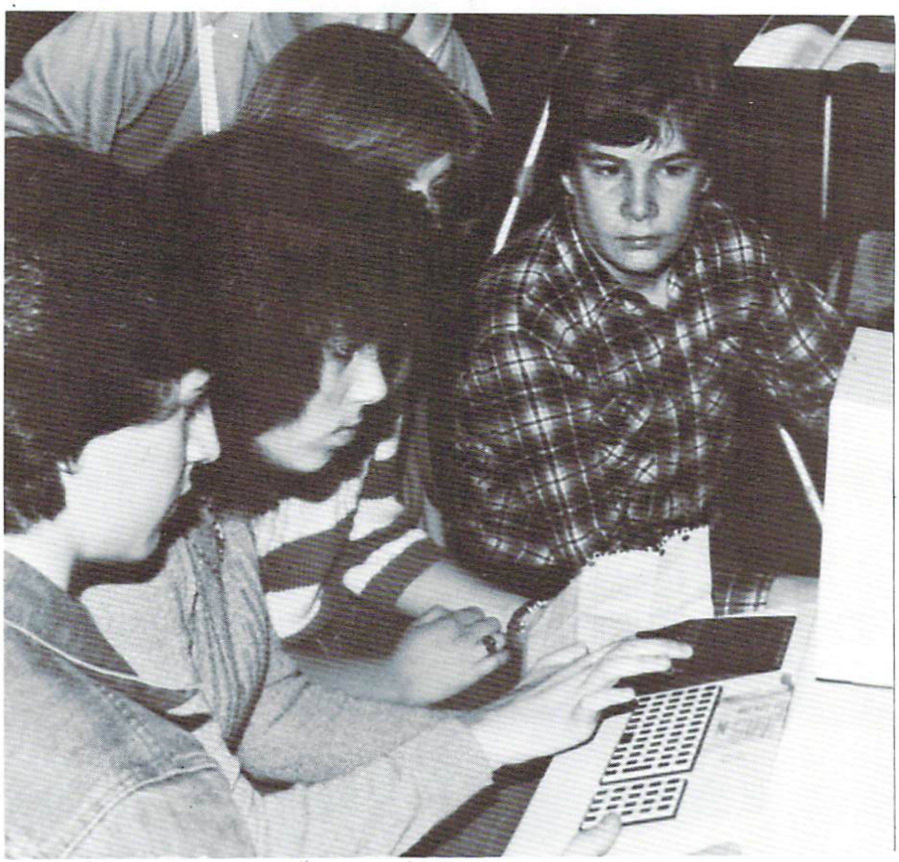
Title: PET INTERFACING
Author: J. Downey/S. Rogers
Price: $16.95

Reprinted by permission of
E-ITV Magazine

How a high school video production department uses the PET computer as a titler.

Bayside's PET Titler

by Arnold Friedman



Students Mike Farrell, Laura Kahn, Carolyn Levine (hidden), and Andy Silibovsky at the PET keyboard, entering data for titling.

A recent article in E·ITV described how a high school in New York City uses a TRS-80 computer as a titler (October issue, 1981). We do the same thing, and more, at Bayside High School with a Commodore PET microcomputer. The difference is that we do not shoot the characters off the monitor for insertion into our video shows.

In our application, the PET provides an in-line video signal which can be fed into any unused video input (camera or auxiliary) of a special effects generator, or into a video mixer. This permits the characters generated by the PET to be keyed in or superimposed, as desired, with very clean results.

The hardware needed is a Commodore PET or VIC computer, and a video buffer board which is available from any Commodore dealer and costs about $20. This board converts the computer output into a composite video signal. Normally it is used, along with a coaxial cable, to interconnect the computer with a classroom television monitor so the computer's output can be displayed on the screen.

At Bayside High, we have a four-camera, monochrome studio, complete with control room, SEG, audio mixers, and an intercom system. Students who are interested in learning television production sign up for TV Studio I, the beginners' course, followed by Advanced TV Studio. Both classes are generally oversubscribed. Within the limitations and restrictions of large classes and short periods, the students write and direct their own original shows, ranging from commercials to documentaries.

For each production, crew positions are rotated to provide everyone with experience as a camera operator, audio engineer, technical director, etc. The most popular crew position is that of graphic "artist"—the person responsible for the characters generated on the computer.

The graphics we generate on the PET are titles in large print, moving messages that crawl across the bottom of the screen, names keyed into the lower third, and credits. The programs that

govern the creation of these graphics are on a cassette, and are first loaded into the computer. Once programmed, the student uses the computer keyboard to make the desired entries, depending on the needs of the particular show.

All of our shows end with the credits for the crew. The program for this, once loaded, requests the name of the person who corresponds to a stated crew position. This information is typed in, and all the names are entered into the computer memory. Upon a signal from the director, the title of the position with the crew member's name underneath, all centered, is fed out from the memory into the auxiliary line. The technical director, at the discretion of the director, then mixes (i.e., keys or supers) the images. The crew positions can be flashed individually, by using either manual or automatic (via pre-specified timing) control.

Other computer programs enable us to key in names of people at the bottom of the screen, run bulletins across the screen with a continuous left-to-right crawl, and display large-letter titles. We can load all of these programs into the computer at the same time, then call up each one individually as needed.

Some of our computer programming students are currently working on the development of a program that will provide the crawl with a controllable speed. We are also experimenting with computer-synthesized sound effects—laser sounds, ringing bells, outer-space sounds—to add to our productions. The audio line comes out of the same video buffer board, and is fed into an audio mixer.

Probably the most exciting thing about using the computer in the studio is the constant discovery of new applications. This amazing piece of hardware has improved the quality and impact of our productions more than any other single unit, and at a fraction of the price.

The double-size titles used to introduce programs.

PET 'BULLETIN' PROGRAM

```
10 DIMA(25)
20 REM PROGRAM TO DISPLAY A CONTINUALLY MOVING
   'BULLETIN'
30 REM ACROSS THE BOTTOM OF THE SCREEN
40 REM PROGRAM FOR COMMODORE PET
50 M$="
60 SI$="WBAY-TV----NEW YORK"
70 FORA=1TO25
80 A$(A)=""
90 NEXTA
100 PRINT"▯TYPE IN YOUR MESSAGE"
110 PRINT"END WITH '..'"
120 PRINT"TO RUN THE MESSAGE, WHICH WILL
    CONTINUALLLY REPEAT, PRESS ANY KEY!"
130 FORA=2TO25
140 INPUTA$(A)
150 IFA$(A)=".."THEN240
160 NEXTA
170 PRINT"? OUT OF ROOM"
180 PRINT"DISPLAY MESSGE AS IS THEN CONTINUE";
190 INPUTA$
```

```
200 IFA$="YES"THEN C=1:GOTO240
210 IFA$="NO"THEN C=0:GOTO240
220 PRINT"YES OR NO"
230 GOTO 180
240 PRINT"]QQQQQQQQQQQQQQQQQQQQQQQ";
250 GETE$:IFE$=""THEN250
260 PRINT"S"
270 FORA=1TO120
280 PRINT" ";
290 NEXTA
300 PRINT"SQQQQQQQQQQQQQQQQQQQQQQQQQQ";
310 A$(1)="* * * *"+SI$+"* * * *"+
    STR$(VAL(MID$(TI$,1,2)))+":"+
    MID$(TI$,3,2)
320 A$(1)=A$(1)+">>>>"
330 FORA=1TO25
340 IFA$(A)=""THEN420
350 FORB=1TOLEN(A$(A))
360 M$=RIGHT$(M$,39)+MID$(A$(A),
    B,1)
370 PRINT M$;"]";
380 FORC=1TO30:NEXTC
390 NEXT B
400 M$=RIGHT$(M$,39)+" "
410 PRINT M$;"]";
420 NEXTA
430 FORA=1TO40
440 M$=RIGHT$(M$,39)+" "
450 PRINTM$;"]";
460 FOR B=1TO30:NEXTB
470 NEXTA
480 GETA$:IFA$="E"THEN 510
490 GETD$:IFD$=""THEN300
500 GOTO 480
510 PRINT"]";:END
EADY.
```

Video adapters formerly used with 9" screen machines will not work directly with the new 12" machines as the polarity of the video out and horizontal sync signals have been reversed at the user Port. Correct this with:

POKE 59520, 12 : POKE 59521, 0

PET People in Prior Lake

Westwood Elementary School, located in Prior Lake, Minnesota, is the largest user of PET computers of any school in the state. According to D.A. Fosse, principal of Westwood, "Our program is unique from the standpoint that all of our teachers are proficient in using the PET. Teachers do not send *children to the computer room, they* take *them and then serve as instructor.*

Presently, Westwood has thirty-one teachers involved in a computer programming course being offered in their computer room by Mankato State University. In the following article, Fosse explains his school's approach to microcomputers in education.

Take a dozen or more computers, an enthusiastic media director, "lotza" software and combine them with an interested staff, excited kids, and supportive parents and you have all the "makins" for a computer program.

If you are faced with the difficult task of attempting to implement an effective computer program in your school, you are among a majority. Most people I have talked to will say, "If only we had more computers."

At this time last year, Westwood Elementary School had one Apple II computer and an enthusiastic media director. She worked to create an interest among staff in the use of the Apple as an educational tool. The general feeling that emerged was, "What can we do with one computer and 756 K-3 children?" To give each child one hour on the Apple would require approximately 150 days. What we needed was more computers. Who wants to ask the School Board for a dozen? We did!

As a member of the state Title IV Advisory Council, I had the opportunity last Spring to witness a computer demonstration presented by the directors of Title IVC projects in Robbinsdale and Morgan. The computers they were demonstrating were not Apples but Commodore PETs. I was intrigued by the capabilities of the PET and by the enthusiasm of the presentors. My interest rose higher when they informed us that three PETs could be purchased for the price of one Apple!

Several discussions, meetings, and visitations later the decision was made to purchase three PETs. These were used during summer school and proved to be so well accepted that we decided to expand. We now have a Computer Room connected to our Media Center, 13 PET Computers and 150+ programs. All classroom teachers, Special Ed. teachers, and aides have been in-serviced and are qualified to use the PET.

At the present time, we have eleven computers in our computer room and three computers on carts for use in individual rooms. Teachers bring their entire class to the computer room to work on the computers. With two children on a computer, everyone gets hands on experience for the entire period. The teacher serves as instructor, monitor, or trouble shooter. The PETs are so easy to use that the children have adapted easily. As children demonstrate proficiency in the use of the PET, they are issued cards qualifying them as our "PET PEOPLE.". They are now entitled to independent use of the PETs. Plans are to place some PETs in the public library during vacation period so children may continue to use them.

All of our programs are stored on regular cassette tapes rather than floppy discs. Cassettes are inexpensive, easy to use, and are very familiar to children, teachers and parents. If you want a duplicate program, simply dub it on a blank cassette.

A three year K-12 plan for projected computer needs has been presented to the Prior Lake School Board. With their support we plan to purchase several more PETs for use in our schools next year. By going with the PET computer I predict that we are 2-3 years ahead of where we might be if we had decided to purchase Apples. Also, no matter how sophisticated computers become in the next few years, our PETs will always be a useful teaching device since they are predominantly used for assisting children with learning skills.

The greatest advantage of the PET remains its cost. You can purchase 10 PETs at a cost of less than $6000.00. This same amount would allow you to purchase only three Apples. To have an effective computer program for all children you need to have several computers. Our computer room at Westwood is real and it is working. Just give us a call at (612) 447-2178.

Commodore Education Shows a Success

Full Schedule Planned for the Fall

by
Jim Bussey

Two Commodore education conferences in the West were a great success and have led to the scheduling of a full slate of shows for the Fall of 1982. Over 700 teachers, principals and district level administrators attended the two shows—one in San Francisco on January 20 and the other in Los Angeles on April 21. Both shows were well received and Commodore has been besieged by educators for more of the same.

The seminars were different in format but they both succeeded in providing educators with the information they need to make decisions about implementing computers in the total education process, both as instructional tools and administrative aids.

The show in San Francisco was composed of two half-day sessions, two speakers and an hour of software viewing at each session. Bill Finzer from San Francisco State University was the keynote speaker. Finzer shared the work that he and his colleagues, Jose Guttierrez and Diane Resek, have been doing with Commodore computers at the Center for Mathematical Literacy at San Francisco State. Funding for their research was provided by the National Science Foundation and the National Institute of Education.

Finzer left his audience with the main thought that the role of the teacher is to bring to life or to animate the great ideas of our culture. We are finding that the computer is a powerful tool in the implementation of that process.

The Los Angeles show took on a different format. The one day conference was divided into three 45-minute seminar sessions, "hands-on" workshop, vendor displays, a keynote address during a catered lunch and a drawing for a free computer for a school.

Glenn Fisher, Computer Coordinator, Alameda County Department of Education, and nationally known leader in educational computing, led well-received sessions entitled "Getting Started with Microcomputers in Schools," "Managing a Computer in the Classroom," and "What Administrators Need to Know About Computers in the School." Fisher's keynote address during lunch was on "Wise Uses of Computers in Education."

John Hosmon, Director of the microcomputer lab at Overfelt High School in San Jose has used 36 PETs in a networking system over the past two years. Hosmon presented sessions on such interesting topics as "Timesharing for the Gifted and Remedial Students on a Network System," "Utilizing the Microcomputer Lab in the High School: C.A.I. remedial math and English, group simulations, teaching programming and advanced programming, teaching computer literacy and word processing," and "How to use Title I computer programs in your school."

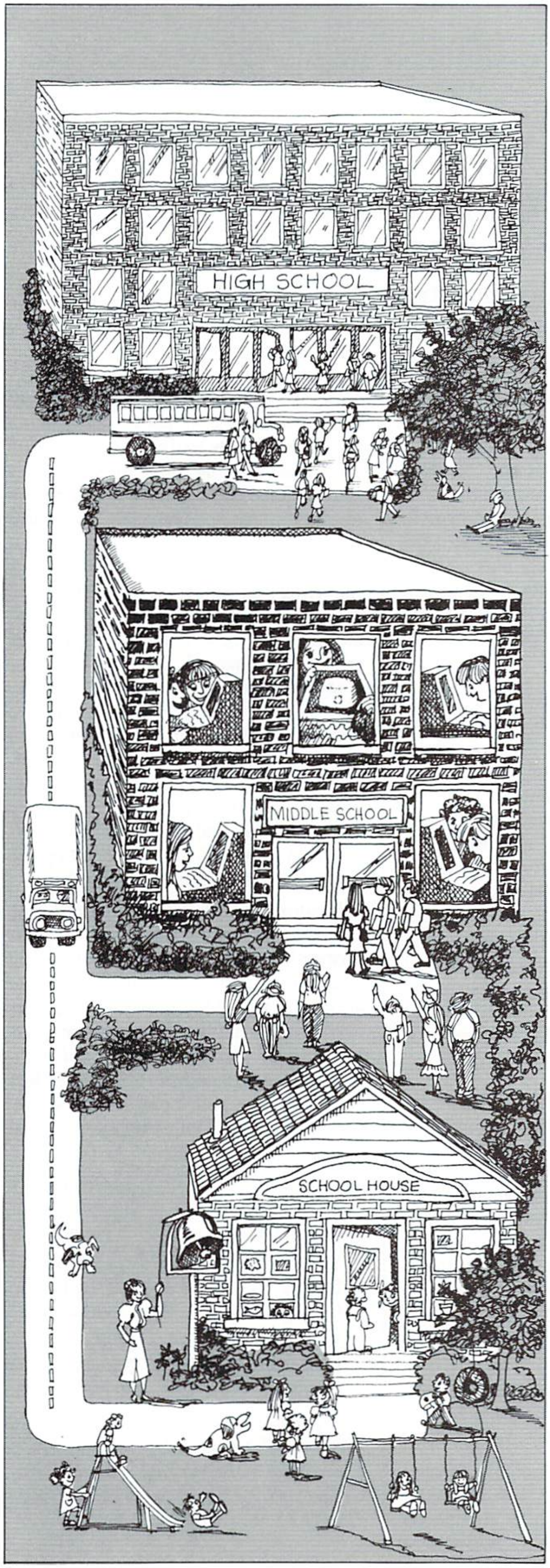
Stanley Greenspan, director of career and vocational education for the San Ramon Unified School District, led workshop sessions on private and government funding for microcomputers. This school district through grant applications, both private and government, and parent-teacher organizations has acquired over 260 PET computers and numerous peripheral devices. Greenspan explained how computers were being used in his district and how they had dramatically improved the educational program in many significant areas for students.

Lou Gonzalez, vice principal of Perris High School in Perris, California led a workshop session on a pilot attendance project being done on Commodore computers through the California State Department of Education. This district has increased its daily attendance from 79 percent to 88 percent since they started keeping attendance records on microcomputers. Gonzalez presented valuable material on how to get started with a microcomputer in attendance reporting. In addition, Jorge Bendick from Human Development Training explained a new counseling program called "Choices," developed by the Canadian government and adapted for use in the United States. This is a sophisticated career counseling program that provides a wealth of information to high school students making choices about careers. Diane Burrup from Educational Development Corporation conducted a workshop in computer management instruction.

Educators at all levels are interested in the applications of computers to their professions. Commodore shows are an important way to become informed about the vast potential that computers hold for the school environment. It is because of this demand for information that Commodore's Western Region has scheduled nine more shows for Fall 1982. The schedule is listed below:

Denver	September 15
San Francisco	September 22
Los Angeles	September 29
San Diego	October 13
Seattle	October 20
Orange County	October 27
Salt Lake City	November 10
Phoenix	November 17
Portland	November 30

For further information, contact Jim Bussey, Educational Consultant, at 3330-B Scott Boulevard, Santa Clara, California 95050, or call him at (408) 727-1130.



Computer Programming in the Middle School

by
Diane G. Benedict

Last spring I plunged into organizing a computer literacy class at Orchard Lake Middle School, a sixth through eighth grade public school in West Bloomfield, Michigan. I was given tremendous support and encouragement from parents, but more importantly from the principal, Robert R. Riggs.

Reading many articles on computer education, I found information available on the purchasing of hardware with many hypes on a variety of software. A huge gap appeared in the area of curriculum and age group exposure. This is an area that today still needs adequate documentation, so that mistakes can be minimized and more educators will feel free to begin programs of their own. Computers are not like so many other educational waves that simply have gone away with time. Most colleges and universities are requiring at least one course before graduation for such fields as engineering, business, economics, and architecture. Graduate students soon discover a basic knowledge of computers can speed their research; robotics is a reality; cottage industries are building in the West and soon will flow across the rest of the country. The potential for this industry has only begun to be explored. So with several courses of BASIC, 13 years of teaching and lots of faith in kids, I decided to begin one step at a time.

STEP 1 — Hardware Acquisition

Searching for and purchasing microcomputers for a classroom of 30 students without endangering the financial standing of the building budget led us to accept a 3 for 2 offer from Commodore Computers. If NASA could use Commodore, we figured that we could as well! Six PETs with graphic symbols plainly displayed on the keyboard provided added security for the neophyte. Due to cost we were forced to purchase cassettes. On second thought, I would definitely forego the cassettes, save the money and purchase a disk drive when able. We added a 19'' Sony B/W monitor which buffered to an old 2.0, 9'' screen PET we had from a year ago. This would permit viewing for students not at the individual micros and could be used for large group demonstrations. The 2.0 was upgraded to 4.0 and the cassette was hooked up to the tape 2 outlet.

A large classroom was assigned to fill the needs of the computer class, as well as a mathematics classroom. Four large rectangular tables were placed against two walls, three tables each holding two computers and one table ac-

comodating the master computer and monitor. Student chairs were scrounged from around the building and were placed so that the programmer would be facing the wall avoiding some distractions and permitting others to watch the ongoing progress. "Tables" of four desks were set up in the center of the room to ease space and create a helping atmosphere during classtime. The teacher's desk and aide table were placed by the window.

STEP 2 — A Philosophy

We decided upon a philosophy of study for the 6th grade students who would be in the required 10-week mini-class for five days a week, 45 minutes each day. As in every classroom we anticipated a wide range of talents and computer exposure. The final decision was to create a serious, exploratory course, challenging each student to his/her ability, yet not discouraging the students who, at a later time in life would awaken to computers as a useful tool for mankind, not simply as convenient home arcade. Of course, we were secure in the knowledge that "talking" to the computer would be a welcome contagious condition and possibly lead a few students to independent creation of games followed by inquiry into more advanced programming techniques. We were convinced that once one of our students wrote a working program, a real pride of accomplishment would result and further encourage all levels of ability.

STEP 3 — Teaching Materials

Creating the actual program became more of a complication than selecting, purchasing and setting up the hardware. Purchasable, applicable materials were non-existent, incomplete or too advanced, which led me to create my own consumable student workbook for machine familiarity, programming, and BASIC working knowledge. Immediately, I discovered a real need for companion tapes, which I wrote and produced during the first session. Various filmstrips and movies were ordered from the county central school district to enrich study. I soon found that ordering the same films for each 10-week session did not work. A substitute was needed for the students' study of computers in society, the work place, basic engineering, the binary system and such. Necessity being the mother of invention, I turned to the light switch on the wall and the three rows of fluorescent lights on the ceiling to explain binary digits, bytes and Assembler. Turning the lights on and off correlated directly to the on-and-off of the electricity of the machine's translation of a programming language to machine language.

STEP 4 — A Total Lesson Plan

Each student was given a syllabus of the course containing due dates, hopeful accomplishments, and credit value of each assignment. These requirements included a Dictionary of Terms made from loose-leaf paper, three letters of the alphabet to a page and one paper fastener. This helped reinforce the new vocabulary as we discussed terms, read articles and began listening to TV and radio more closely. Whenever a new word appeared, the child would list it in the dictionary and when the class was together as a group, definitions were discussed. Required terms included such words as hardware, software, byte, input/output, disk drive, syntax, debug, etc. A scrapbook was also required to further broaden their exposure to computers and their wide usage today. The scrapbook, which was the only homework assignment for the 10-week session, contained information on Kinds of Computers, Computer Languages, Computer Jokes and Commercials, Articles on computers, handouts, a worsheet on the computer system, types of I/O, notes on the films, and a bibliography.

Other class assignments included discussing the history of computers from the earliest counting of pebbles to the computation machines, the revolution of industry, the advent of computers 20-30 years ago, and now the personal computer industry of the past 10 years. A fiction story was written to accomodate students whose talents encompassed the challenge of using as many computer terms as possible. Using *"Be a Computer Literate"* by Marion J. Ball and Sylvia Charp, the classes took notes on the uses of computers in various arenas of life. Additional applications were added from queries made of parents and teachers. None of the written work was long and tedious. Exposure was the objective. A secondary advantage concerned the problem of having a substitute for a computer classroom. These written assignments fulfilled both needs, added learning and substitute plans. In addition, the material was worked on whenever the computers were being used and the student could not try out a program. On-line computer time was arranged on a 20 minute basis. A change of users still allowed for five minutes to return folders, stack chairs and shut down the equipment. To do any SAVE-VERIFY's a 10-minute warning was given. To facilitate assigning whose turn it was to use the PETs, I had the class number off by 4's, providing two people per unit. This way posted numbers would signify user time. The student's policed themselves quite adequately.

Using my workbook, and the companion tapes, we learned BASIC programming commands. The extent of the lessons was once again dependent upon the class's aptitude. Assignments were given after the introduction and experimentation of new commands. REM and CLEAR SCREEN were required for each program. REM told me which assignment was being handed in and by whom. CLEAR SCREEN allowed the novice to see the RUN without LIST interference on screen. The first lesson on the PRINT statement, we literally wrote together. Each programmer went to a unit and checked to see if the OUTPUT was the desired result. I

quickly learned that lack of typing skills was a problem. So I laminated 30 keyboard reproductions and passed them around for practice. Styles varied from one finger to a ballet of crossed arms; the latter was the only action I discouraged. INPUT, IF THEN and FOR NEXT were definitely covered each session. Each with an assignment following the introduction, practice lessons and encouragement.

For variety we would have fun days with graphic games of exploration. *"PET BASIC Training—Training Your PET Computer"* by Ramon Zamora, Robert Albrecht and William Scarvie has excellent suggestions and programs for this purpose. Also, demonstrations were given by the Computer Resource Team (CRT), a group of eighth grade computer whizzes. Having these 8th graders in and out of the room all period challenged and encouraged the class to explore further programming techniques.

WARNING: Teaching computer programming to children can be dangerous—it is very addictive.

STEP 5 — Evaluation

After each 10-week session I evaluated the program for clarification and modification. For example, the first session was with the incoming 6th graders who were overwhelmed with a new school, a new program to follow and a new class dealing with the mystical computer. I expected too much from this group for their age and readiness. By the second session the program ran much more smoothly and realistic expectations were met on both sides. Plus, most of the students came to the class hearing how much fun it was.

The entire idea has caught on and the positive pressure from parents and students in the 7th and 8th grades has created a change in scheduling for the next year. Next year we are scheduling Computer I, Computer II and Advanced Computers as elective choices for the seventh and eigth grades. Three days a week I remain at school for an hour or so, allowing the interested students more computer time. The exchange between these kids is marvelous, providing an excellent lesson in sharing knowledge and exploring ideas as a team. We even get a few "I just happened to be in the neighborhood" boys from the high school So I am quite confident the lack of structure will more than adequately work.

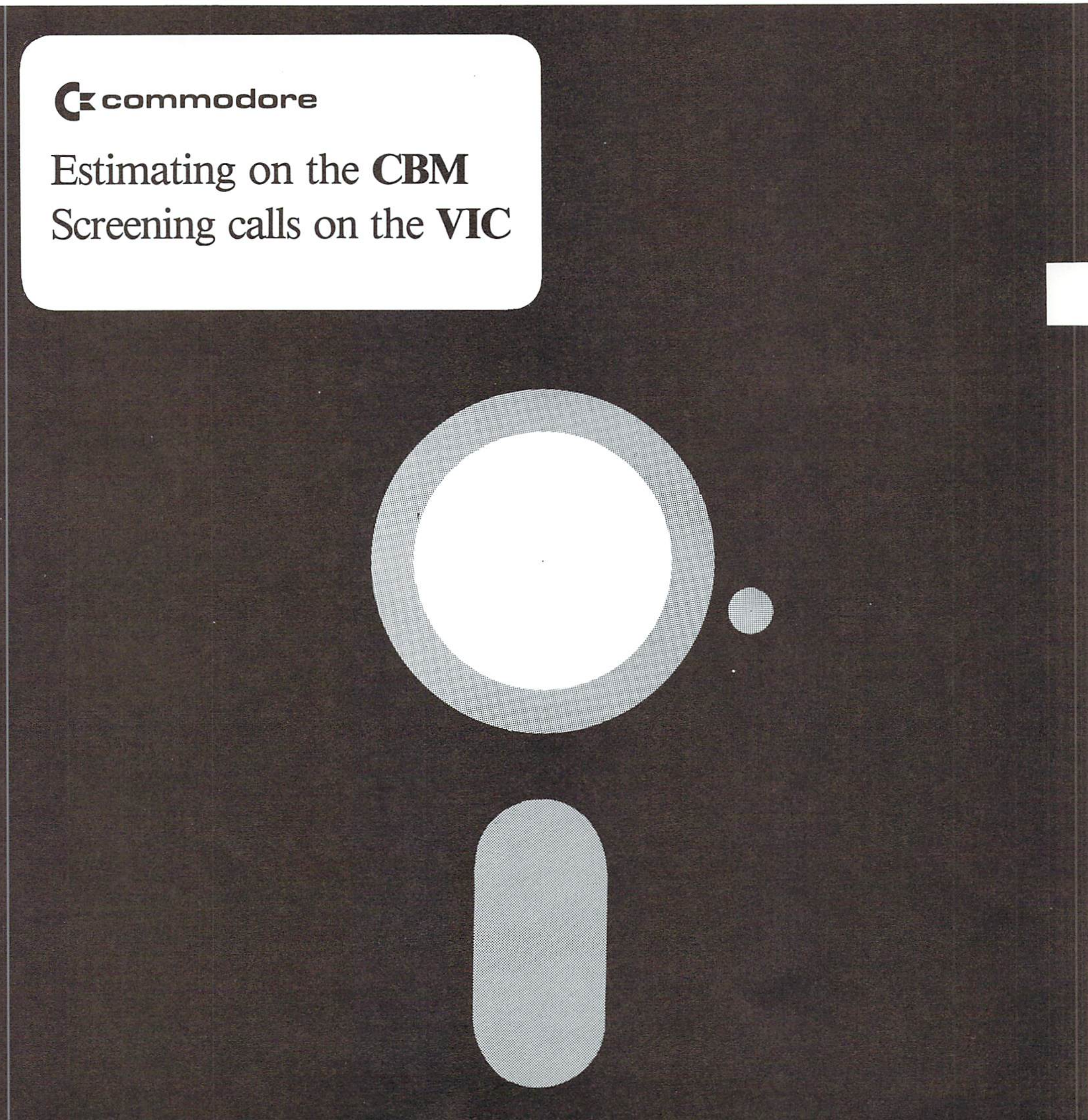
NOTES:

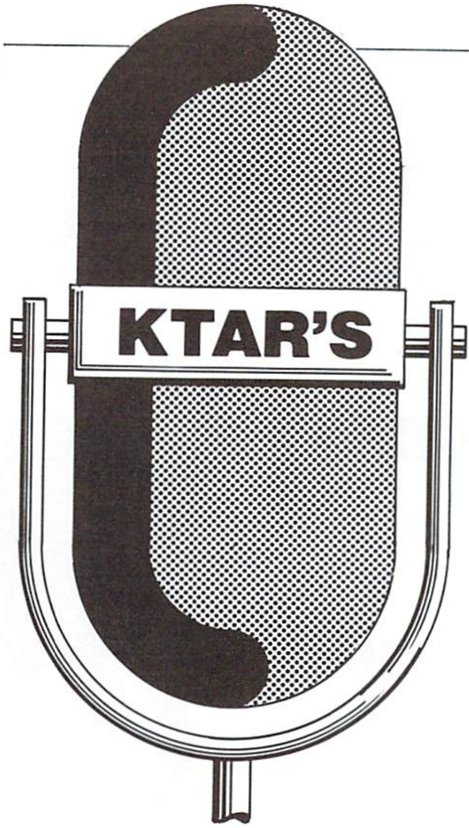
One of my favorite reactions from the class involves communication with parents. The subject being so new places everyone on an equal footing. In fact, one girl was delighted that she could finally convince her father that she was able to write a program for his personal computer. He was impressed, and they have since sat down and worked together with the machine.

I should not forget to tell you about our user-friendly approach with the students. After a much anticipated contest, the PETs all received names! Snoopy, Ralph, Morris, Radar, Woodstock, Merlyn and Reservit (a take off on reserved, one machine kept for advanced programmers in the class) and of course the Commodore, our monitor. It is so much easier to say "Morris, you forgot to stack your chairs" or "Snoopy is available if anyone needs a computer."

I am ecstatic with the past year and am looking forward to helping other educators create a computer program for the students, geared to both the teacher's and student's benefit. This is a real opportunity to learn from your students and not be entrapped by the educational syndrome of only imparting information over and over again. I have a love of learning and feel this program is an honest exchange between minds and talents disallowing the categorizing of body, age, sex or authority. The machine is only as capable as its programmer, so we are all at primer level. Next year we'll be blue birds headed for the time when we can fly with the eagles.

BUSINESS





Talk Screen Program

Ever since the advent of commercial radio over 60 years ago, the medium has survived the coming of more sophisticated and visually pleasing means of entertainment such as motion pictures, television and video games. Now, the latest challenge—personal computers—has arrived.

Assuming the role of trendsetter, a Phoenix radio station has already hopped aboard the microcomputer bandwagon. Using Commodore's low-priced VIC 20 computer, station KTAR-AM has implemented a caller-selection process labeled the "talk screen program." Designed by Tim Greer, an announcer on KTAR's FM station, the program aids the station in screening calls for over 15 hours of locally-produced talk shows per day.

Normally, a producer or switchboard operator answers the phone and puts callers on hold until the talk show host—working without knowledge of who the callers are or what they might say—puts them on the air. Obviously, the host is often grasping at straws, unable to anticipate or prepare for the caller's comments until it's too late.

Now, with the VIC 20 and the talk screen program, the KTAR switchboard operator records a caller's age, neighborhood, and topic for discussion. In addition, the operator identifies the phone line the caller is on and the time the call was put on hold. The information, for up to six callers at a time, is displayed on the host's TV monitor. In fact, the station's VIC 20 is connected to three separate monitors. One monitor is used by the producer, the second is located in the main call-in area, and the final monitor is found in the sports call-in studio.

During KTAR's sports programs, an announcer can get a valuable jump on listeners' questions. If a caller has a specific question on a sports statistic, the show's host can see the question on the monitor, turn to a resource book, research the material, and have the information ready BEFORE the call is even answered.

While some feel this type of screening can ruin the spontaneity and originality of a talk show format, Greer disagrees. "The station would love to get as many viewpoints on a subject as possible," he explained, "But we are fighting ratings battles. We want to keep the shows interesting, light, lively and moving. The talk screen program allows the host to achieve that goal."

In addition, the switchboard operator can control the flow of calls. "For instance," said Greer, "four people may call the station with almost the identical question or comment. We can now advise the caller that their question or comment will be addressed shortly, and ask them to hang up and listen for the response. Callers no longer have to be put on hold unnecessarily. And, too, we avoid on-the-air duplication."

Greer feels that the system allows talk show hosts to maintain varied programs where the appeal of calls can change the way a disc jockey would rotate different types of music. Greer pointed out that the radio broadcasting industry is naturally governed by time constraints, which could be handled more efficiently using the talk screen program. Aside from displaying general topics on the screen, the producer can send important "reminders" to the announcer. Messages such as 'LET'S WRAP IT UP' can serve as simple but important directives. And, because the screen contains a clearly visible real-time clock, the announcer is always aware of the time remaining. Even if the announcer should lose track of the time, the VIC will flash the valuable "time remaining" in reverse video. Easy-to-read messages are also displayed in brilliant VIC-generated colors.

Another reason for easy readability is the VIC's 22-character screen format. "During a talk show," Greer pointed out, "the host is totally absorbed in conducting an interview, yet still must contend with handling calls, talking to the guest and coordinating news feeds as well as commercials." With the 22-character format, the screen displays a minimum amount of information that is very easy to read. The announcer can see just the information he or she needs, without losing control of the program. "It's the perfect compromise," says Greer.

Greer's system is not the only automated call selection program in the country. Radio station WIND-AM in Chicago has a custom system designed to operate an Apple computer. However, Greer eagerly points out the cost effectiveness of his VIC system versus the $5000 Apple counterpart. "We could have a $5000 dedicated system set up to do what the VIC does. But for the limited amount of time that the system would be used, compared to its

capability and capacity, the system far exceeds their needs. Using a $5,000 system to do what our $300 VIC 20 can do is a ludicrous use of computing power.''

How did Tim Greer, radio announcer become Tim Greer, programmer? ''Believe it or not,'' said Greer, ''for years I was completely turned off by computing. When I finally decided to make the commitment to personal computing, it didn't take long for me to decide on the VIC 20. I was impressed by the price/performance ratio as well as the color capabilities. And I knew the VIC would allow me to ease into computing.

However, shortly after acquiring his VIC 20, Greer quickly went from upstart home hobbyist to serious programmer. ''One day,'' Greer recalled, ''our general manager approached me with the task of creating a program that would allow us to automate our call selection. In no time at all I became possessed with my task. If the VIC 20 is the ''friendly computer,'' then I must be the walking, talking, moving example of user friendliness.

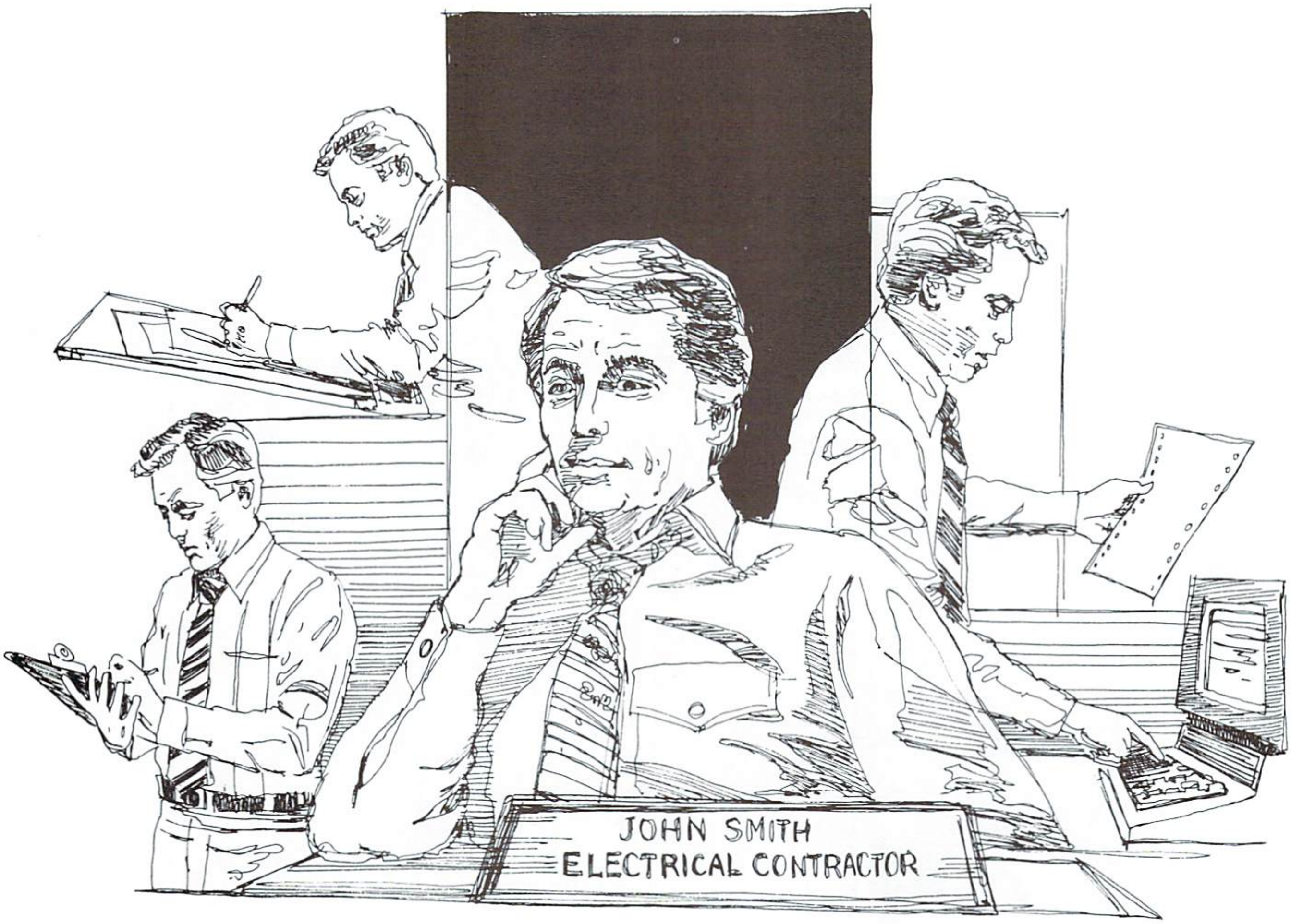
''Just by following the VIC reference manual, and receiving some limited technical direction, I was able to develop the talk screen program. I was so comfortable using the VIC I felt like I must have been a programmer in a previous life.''

Although Greer intends to market his product very soon, he stressed his desire to thoroughly debug the program. Still, Greer is encouraged by the dependability of his product. ''Since February, 1982, when the talk screen program was fully implemented at KTAR, there has not been a single error,'' Greer happily asserted.

The decision-makers at KTAR are also planning to use the demographic information compiled by the system to more closely determine types of callers and topics of interest—which will hopefully provide better quality programming.

Sixty years ago, what radio pioneer would have considered an innovation like a computerized caller-selection program. And, with even more technological advances still to come, who knows what surprises are in store for radio listeners in 2042!

–Paul Fleming



CBM Rescues Electrical Contractor

In 1977, even before "Reagonomics," the U.S. government had already initiated a plan to cut down on expenditures. For a company that works almost exclusively on government contracts, cutbacks can be a death blow.

Lovell Electric of Franklin Lakes, N.J., was one of the many companies affected by the cutbacks. Lovell Electric is an electrical contracting company that designs and builds electrical projects for the government. Theodore Lovell, executive director of Lovell Electric, admits his company was in a bind.

"To stay alive we had to lower overhead," said Lovell, "and the only way to lower overhead was to increase efficiency.

"I had this idea, based on an article about microcomputers in *Popular Science* and an ad from Commodore," he said, "that a micro could replace some of our personnel." Lovell took a risk and became a "pioneer" by buying one of the first personal computers, an 8K PET. It had to be ordered by mail directly from Commodore's factory and had a serial number under 1000. Back then Lovell had three employees in his office, a general secretary, an executive director (Lovell), and an estimator/engineer. "The payroll was what was keeping us from being as profitable as we should have been," Lovell said. The secretary and the estimator engineer were both planning on retiring. Lovell could either hire people to replace them or find a more efficient solution, but his quest for a solution was not an easy proposition. "Initially," he recalled, "our first computer was a disaster. We couldn't find the switch to turn it off, and every time we put something on the screen we got a 'syntax error.'"

Business software was almost nonexistent in those days and Lovell had no programming background. Many people would have quit, but he persisted. With the help of a knowledgeable friend and a copy of Coan's "Basic Basic," Lovell's $695 investment began paying dividends. "By March 1978 we had an effective invoice system up and running, and, for the first time in 25 years, we could quickly determine our exact cash position," he said.

Encouraged, Lovell began to attack other problems. He started by writing a program for accounts payable. Next, he wrote an accounts receivable program. Then, with the acquisition of a disk drive, he wrote a payroll program. Accounting problems with payroll and payables virtually ceased for the firm. In addition, he has also written a motor design program that cuts his engineering time in half.

Lovell's programs have been streamlined and perfected over the years. His payroll program, for example, had its

special challenges. Lovell employs from five to 18 people, depending on the company's current contracts. Each employee can earn up to six different hourly rates per day. His payroll program calculates all the different pay rates, adds in bonuses and then prints out checks.

Today, although business application software is plentiful, Lovell still uses all his earlier programs, but has added Wordpro 4. The reason is that he believes that software is better if it is tailor-made.

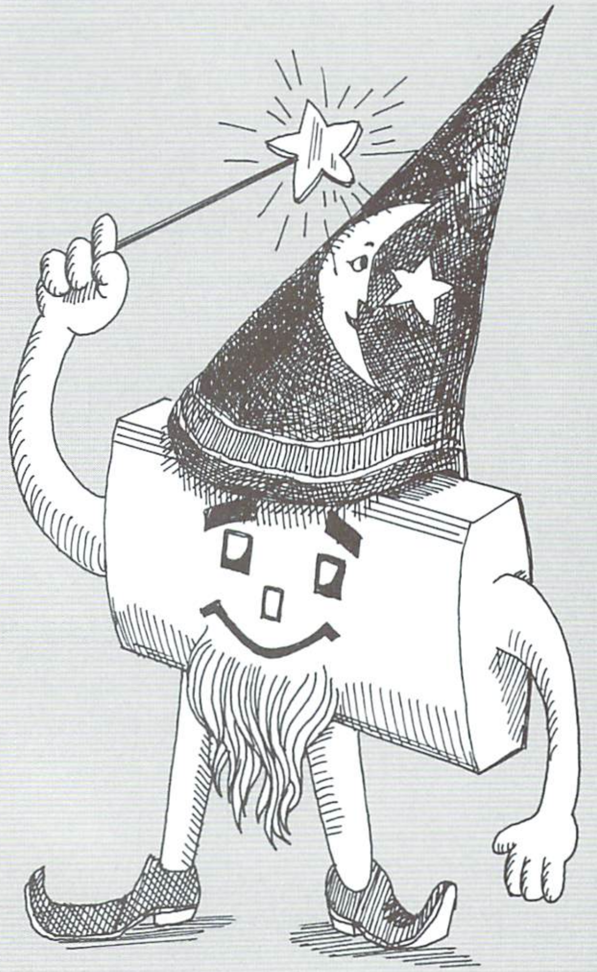
"No two companies need exactly the same kind of programs. I started with no computer background and learned as I went along." He is now somewhat of an expert at programming and strongly recommends to anyone using a micro for their business to "learn how to program or have a programmer in the organization."

Lovell's entire operation is computerized. His payroll is also used for reports that he must to submit to the state government. His records are no longer done by hand and state auditors have never questioned his reports. His clients are also impressed with his fast and professional estimates that he can produce using his estimate engineering program and the computer's word processing capabilities.

Lovell's enthusiasm for micros—and Commodore computers in particular—is contagious. He has helped several friends and associates get started with systems. As president of his local school board he has helped his school district to computerize. Lovell convinced other board members to take advantage of Commodore's "3 for 2" educational grant program. To date, the school board has bought 12 more Commodores.

"But back to business," Lovell says, "we expect most equipment to pay for itself in five years. The Commodore pays for itself every two months. For me, it is the difference between profit and loss. If I sound enthusiastic, it's because I am. If Commodore hadn't come out with a computer in 1977, I don't think Lovell Electric would be here today. Now, we not only exist—we do it better."

— John O'Brien



The VIC Magician
Using Those Mysterious Programmable Function Keys

Michael S. Tomczyk
Product Marketing Manager

Everyone always asks how the VIC-20's programmable function keys work (The function keys are those large yellow keys on the right side of the VIC-20 keyboard).

Special function keys were added to your VIC to let you take advantage of "one-key" programming features normally found on much more sophisticated (and expensive) office computers. It doesn't take much effort to program these keys and that's what the following lesson shows you.

When you first got your VIC-20, you were probably surprised that nothing happened when you pressed the function keys. That's because the function keys are **programmable,** meaning you have to program them to do something before they work.

Before getting into the programming part, let's take a closer look at the keys, themselves.

To begin with, the function keys are numbered 1 through 8. The odd-numbered keys 1, 3, 5, and 7 are obtained by simply typing those keys, and the even-numbered keys 2, 4, 6 and 8 are obtained by holding down the SHIFT key and then typing the appropriate key. This lets you use four keys to get 8 separate functions.

Now, here's an important point . . . each key has a special NUMBER which you must use when programming that key. This number is called the **CHR$ NUMBER.** All of the VIC's keys have CHR$ numbers, which are listed in the ASCII AND CHR$ CODES table on page 146 of your VIC user's guide (or page 273 of the VIC PROGRAMMER'S REFERENCE GUIDE).

CHR$ CODES FOR FUNCTION KEYS

FUNCTION KEY NUMBER	CHR$ CODE
f1	CHR$ (133)
f2	CHR$ (137)
f3	CHR$ (134)
f4	CHR$ (138)
f5	CHR$ (135)
f6	CHR$ (139)
f7	CHR$ (136)
f8	CHR$ (140)

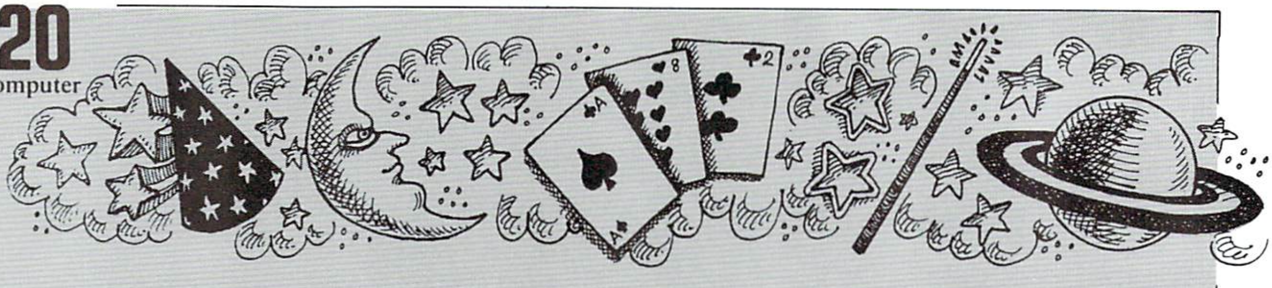
Note that the CHR$ numbers are not in exact order . . . the odd numbered keys are numbered 133–136 and the even numbered keys (which you must SHIFT to use) are numbered 137–140.

Using the GET Statement—"Hit any Key"

Before we explain how everything works, let's write a short program using a function key, to show you how it works. The first thing we do is TELL THE VIC TO LOOK AT THE KEYBOARD. This instructs the VIC to check the keyboard to see if you have pressed a key. The BASIC command for this is:

```
10 GET K$:IF K$ = " " THEN GOTO 10
```

This is called a "GET Statement" and is usually placed as a single line in your program. If you include this line all by itself in your program, the VIC will look for ANY KEY to be hit. So let's write a short program which tells the user to "HIT ANY KEY TO BEGIN." Enter this



program, type the word RUN and hit the RETURN key:

10 PRINT" HIT ANY KEY TO BEGIN":PRINT
20 GET K$:IF K$=""THENGOTO20
30 PRINT"♥ R PROGRAM BEGINS."

(BEGINNERS: be sure to hit the RETURN key at the end of each line to enter it.)

THIS MEANS HOLD DOWN THE **SHIFT** KEY AND TYPE THE **CLR/HOME** KEY

THIS MEANS HOLD DOWN **CTRL** AND TYPE **RVS ON** KEY

See how it works? Now let's go one step farther. After we BEGIN our program, we may want to make the program WAIT until the user is ready to continue. This technique is often used in educational programs when you want to give a student time to study something on the screen before moving on. Hold down the RUN/STOP key and hit the RESTORE key to "exit" your program. Now type these additional lines (the VIC automatically adds them to the program above, which is still in VIC's memory):

40 PRINT"♥ WAIT HERE.":PRINT:PRINT"HIT ANY KEY TO CONTINUE.":PRINT
50 GETK$:IFK$=""THENGOTO50
60 PRINT"♥ R PROGRAM CONTINUES."

Note that this program now uses the same GET statement in two different places to check the keyboard to see if a key has been pressed. You can insert line 20 almost anywhere to make the VIC wait until a key is pressed . . . but if you do this in the middle of a program, don't forget to PRINT a little message telling the user to "hit any key to continue."

Now let's take a closer look at how the GET statement lets us check the keyboard . . . and program our function keys.

More Information About the GET Statement

As already explained, the **GET K$** line tells the VIC to check the keyboard to see if a key has been pressed. There are three important things to remember about using this line in your program. First . . . notice how we used the **string variable K$** in lines 20 and 50 above? Although we used K$ as our variable (K for "Key"), you can use **any legal string variable**, such as A$, KK$, B1$, etc. in this line. For example, you could use: **50 GETRR$:IFRR$=""THENGOTO50.**

The second thing to remember is that the GET statement line **always GOES BACK TO ITSELF.** In other words, if you put this line at line 100, the last part of the line would read . . . **THEN GOTO 100.** This makes the VIC keep checking over and over again until a key is actually pressed. So the GET statement includes a GOTO to its own line number.

Finally, don't attach any other BASIC commands on the same line as your GET statement. For example, don't put a colon and add more BASIC commands to lines 20 or 50 above.

Programming a Function Key

Now let's make a function key do something—how about amending our program above so that it only begins if you type function key 1, and only continues if you type function key 3 . . .

We'll use the SAME PROGRAM we used above and simply modify two lines and add a few extra lines to tell the VIC to accept ONLY DESIGNATED FUNCTION KEYS instead of "any key." To begin, type the word LIST and hit RETURN. This displays your program. Now modify your program by changing or adding the elements shown in BOLDFACE in the finished program below (If you need help editing program lines, see pg. 74 of the VIC-20 PROGRAMMER'S REFERENCE GUIDE, "Editing Lines.")

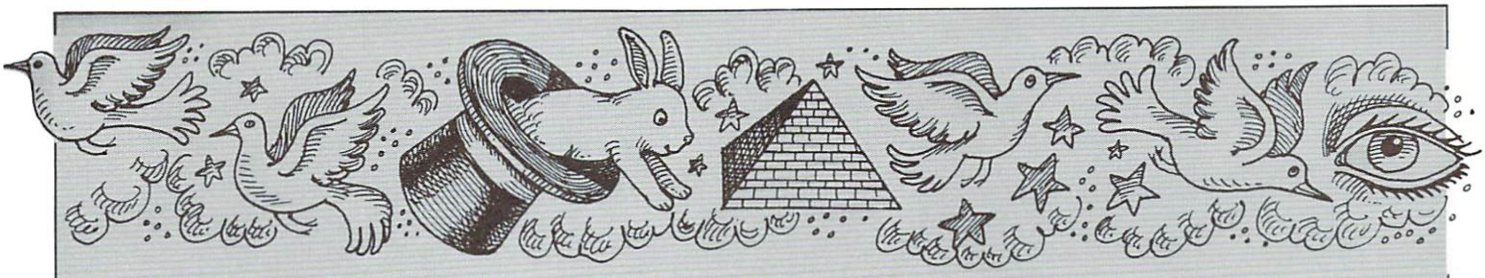
10 PRINT " HIT **THE F1** KEY TO BEGIN":PRINT
20 GET K$:IF K$=""THENGOTO20
25 IFK$〈〉CHR$(133)THENGOTO20
30 PRINT" PROGRAM BEGINS
40 PRINT" WAIT HERE."PRINT:PRINT"HIT F5 KEY TO CONTINUE.":PRINT
50 GETK$:IFK$=""THENGOTO50
55 IFK$=CHR$ (135)THENGOTO60
56 GOTO40
60 PRINT"♥ R PROGRAM CONTINUES."

LINE 25: Line 25 tells the VIC: "If you check the keyboard and find any other key but the F1 key being pressed, GOTO line 20 and check again. The 〈〉 sign means less or greater than (or you might interpret this as "not equal to") . . . in this case, the program treats CHR$ (133) as the number 133.

With this example, your "real" BASIC program would start at line 30. Naturally, you can alter (increase) the line numbers after line 30 to give yourself more room if you're inserting a longer BASIC program here.

CHR$ (133) is the CHR$ number for the F1 function key, and IF K$=CHR$ (133) means "IF F1 IS BEING PRESSED." You can designate a different key by changing the CHR$ number (see chart above). For example, CHR$ (139) changes the key to F7.

LINE 26: This is IMPORTANT! In line 25 we told the VIC to GOTO line 30 if you press the F1 key . . . but we didn't tell the VIC what to do if we DON't press the F1 key. So what happens is this . . . the program will "fall through" the IF . . . THEN statement in line 25 and



KEEP GOING if we don't do something to stop it. The solution is to include the GOTO in line 26 which tells the VIC to GOTO line 10 and keep PRINTing that opening message until the F1 key is pressed. This is important because if you experiment with function keys you may find your program "falling through" the IF . . . THEN line and RUNning "wild." The answer is to insert a GOTO after the IF . . . THEN statement, which keeps the program going back to the prompt message until the proper key is pressed.

LINES 55–56: These lines are the same as Lines 25–26. Line 55 is the IF . . . THEN statement which designates the F5 key to be pressed and Line 56 keeps the program "looping back" to the message in line 40 until the F5 key is pressed.

Programming Function Keys to Perform Functions

So far, we've used function keys to 1) start your BASIC program, and 2) continue a BASIC program in progress. Both of these techniques used the function keys to add some nice cosmetic touches to your BASIC programming, but they really didn't show you how to use the VIC's function keys to perform "real" functions, so that's what we're going to show you next.

Type the word NEW and hit RETURN to erase your previous program, and type in the following program. This is a VIC SPEAKER DEMONSTRATION which lets you type any NOTE VALUE and hear it played on any of the VIC's four internal "speakers." Function key 1 plays the note you entered on Speaker 1 (the lowest tone speaker), function key 2 plays the note on Speaker 2, and so on. Speaker 4 is the "white noise" or "sound effects" speaker.

```
10 PRINT"♥ SOUND DEMO":PRINT:
   POKE36878,15:S1=36874:
S2=36875:S3=36976:S4=36877
20 PRINT"♥ "
30 PRINT"F1 PLAYS SPEAKER 1":PRINT
40 PRINT"F2 PLAYS SPEAKER 2":PRINT
50 PRINT"F3 PLAYS SPEAKER 3":PRINT
60 PRINT"F4 PLAYS SPEAKER 4":PRINT
70 PRINT"F5 LETS YOU ENTER A NEW NOTE
   NUMBER":PRINT
100 PRINT"INPUT A NOTE NUMBER BETWEEN
128 AND 255":INPUTA
120 GETM$:IFM$=""THENGOTO120
130 IFM$=CHR$(133)THENX=S1:GOTO500
140 IFM$=CHR$(137)THENX=S2:GOTO500
150 IFM$=CHR$(134)THENX=S3:GOTO500
160 IFM$=CHR$(138)THENX=S4:GOTO500
170 IFM$=CHR$(135)THENGOTO10
180 GOTO 120
500 FORT"1TO200:POKEX,A:NEXTT:POKEX,0:
   GOTO120
```

Type RUN and hit RETURN.

Now, this little program is just one example of how you can assign different function keys to perform different functions. In this case, we followed some of the general rules we've already established, and introduced a couple of new programming techniques. Let's examine the program, line by line.

LINE 10: contains our "opening message." We also took advantage of the extra space on the line to turn the volume to its highest level (POKE36878,15) and assigned some easy-to-remember variables to each of our four speakers, S1, S2, S3 and S4. Obviously, it's easier to type S1 than 36874, and it also saves memory.

LINE 20 is a blank line.

LINES 30–70 provide instructions for using the program.

LINE 100 is a special INPUT instruction which asks for a NOTE VALUE to be entered into the computer. The INPUT statement assigns the value A to whatever number was typed in by the user.

LINE 120 is our GET statement to check the keyboard.

LINES 130–160 are IF . . . THEN statements which match the messages in lines 30-60. Notice that we could have written EACH LINE like this:
IFM$=CHR$(133)THENFORT=1TO200:
POKES1,A:NEXTT:POKES1,0:GOTO120.
Instead, we wrote a more efficient program which put a "GENERALIZED SOUND ROUTINE" on line 500 so all we have to do in lines 130–160 is define X as the proper speaker and GOTO line 500 to execute the note, then jump back (GOTO) to line 120 to check the keyboard again.

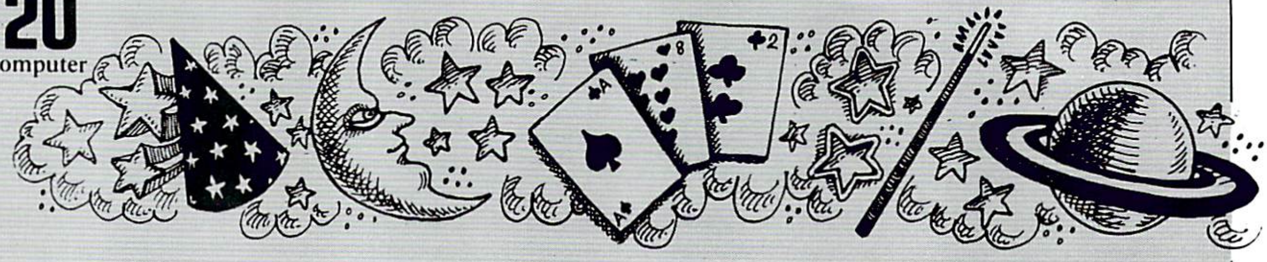
LINE 170 tells the VIC to go back to the beginning of the program and ask for another note value if the user types F5.

LINE 180 tells the VIC to go back and check the keyboard.

LINE 200 contains a time delay loop which specifies how long the note is played when you press the function key (try changing the number 200 to something else). POKE X,A means to POKE the speaker defined as X in lines 130–160 with the note defined as A by the INPUT statement in line 100. In other words, if you entered the number 201 and pressed function key 1, the VIC responds by playing the "D" note from its middle octave. (see your user's manual for note values)

For More Advanced Programmers

There is ANOTHER way to check the keyboard to see which KEY is being held down, using the command **PRINT PEEK (197)** or **PRINT PEEK (203).** You can



PEEK into either of these special memory locations to find out which KEY is being held down. Note that by KEY we mean the physical key being held down, NOT THE SYMBOL. In other words, F1 and F2 are interpreted as being **THE SAME KEY** because we are detecting the physical key is being mechanically held down. The values for the function keys are given below (a complete chart of values for all keys is given on page 179 of the VIC-20 PROGRAMMER'S REFERENCE GUIDE).

VIC KEY	VALUE RETURNED WHEN YOU PEEK (197) or PEEK (203)
no key	64
f1/f2	39
f3/f4	47
f5/f6	55
f7/f7	63

How to use this technique will be covered in a future VIC MAGICIAN, but the general principle involves PEEKing to see if a key is being held down and if the value returned matches the value of a specific key, you make your program perform a specific action. For example, you might write a music program which tells the VIC to play certain notes when certain keys are being held down, and to stop playing those notes when the keys are released. Here's a short program to start you off . . . when you RUN this program, it will play a note when you hit the f1/f2 key. If you get the PEEK values for ALL the keys from the PROGRAMMER'S REFERENCE GUIDE, and match them to the proper note values from the TABLE OF MUSICAL NOTES, you can write your own "VIC PIANO!"

```
10 POKE 36878,15:S1 = 36874:S2 = 36875:S3 = 36876
20 IFPEEK (197) = 39THENFORT = 1TO50:
   POKES1,200:NEXTT:POKES1,0
30 IFPEEK (197) = 47THENFORT = 1TO50:
   POKES2,200:NEXTT:POKES1,0
40 IFPEEK (197) = 55THENFORT = 1TO50:
   POKES3,200:NEXTT:POKES1,0
30 GOTO20
```

The key elements in Line 10 are turning on the volume and defining the "speakers" as S1, S2 and S3. In lines 20–40 we PEEK (197) to see what key is being held down, then insert a time delay loop (1to50) to specify how long each note is held down, then POKE a note value into the speaker we want, then we turn OFF the speaker (otherwise it would keep playing). Change the number 50 to a higher or lower number to increase or decrease the duration each note is played.

Summary

This introduction to the VIC's programmable function keys is only a beginning. You can probably design programs which do much more than those described here . . . for example, you could make the function keys stand for different colors, and instead of POKEing note values, designate special keys to POKE different COLOR COMBINATIONS using the SCREEN AND BORDER COLOR chart in your user's guide.

You could specify each function key to perform a different complex calculation by combining the function keys with DEF FN statements. You might want to INPUT a series of numbers and perform several multi-step calculations which you can execute simply by pressing the function keys.

You might match the function keys to program SUBROUTINES which allow the user to access different portions of a long complex program, so he can jump from a main program into a smaller subroutine, and then come back to the main program . . . all by pressing one or two function keys.

Finally . . . you may have already discovered that programmability isn't limited to the VIC's function keys. In fact, ALL the VIC's alphanumeric and graphic keys are programmable, using the techniques described here in combination with the CHR$ values for each key. To program a key to perform a special function, simply use that key's CHR$ value just like the function key values described here.

If you develop an imaginative application for the VIC's special function keys, drop a letter to THE VIC MAGICIAN in care of our magazine and we'll share your discoveries with the rest of the "VIC world." **Enjoy!**

Two COMMODORE cartridge products which offer built-in function key programming include the SUPEREXPANDER cartridge and the PROGRAMMER'S AID CARTRIDGE. Both cartridges let you redefine the preprogrammed function keys by typing "KEY" and hitting RETURN. See your Commodore Dealer for more information.

Commodore Programs Move into the *Fast Lane* with PETSPEED

Petspeed — The Optimizing Basic Compiler that runs Commodore BASIC 40 times faster. You can dramatically reduce long processing times, tedious disk handling, and long print runs. No other compiler can offer the same speed, compatability and trouble-free compiling as Petspeed.

Compatible —Petspeed compiles any BASIC application and is available for any combination of 4000 and 8000 series Commodore equipment.

Faster The key that sets Petspeed apart from other compilers is **optimization.** Where most compilers merely translate from one language to another, Petspeed analyzes your source program to eliminate unnecessary complexities, thus speeding processing time. Your program is reduced to much smaller components and reassembled into a far more efficient form. Compare these optimizing features:

- 4-Pass compiler.
- Automatically uses faster integer arithmetic when possible.
- Automatically handles frequently occurring variables and arrays.
- Subroutines no longer need be located at the beginning of your program.
- Petspeed automatically calls all subroutines at maximum speed.
- Petspeed runs twice as fast as other compilers.
- Larger programs require far less memory when compiled.

Easy to Use Petspeed is as easy to use as these screen displays illustrate.

Directory BEFORE compilation.

Simply type in your program name.

Directory AFTER compilation

It isn't necessary to add compiler directives. Simply type in the program name. In less than 2 minutes, you'll see your program run up to 40 times faster.

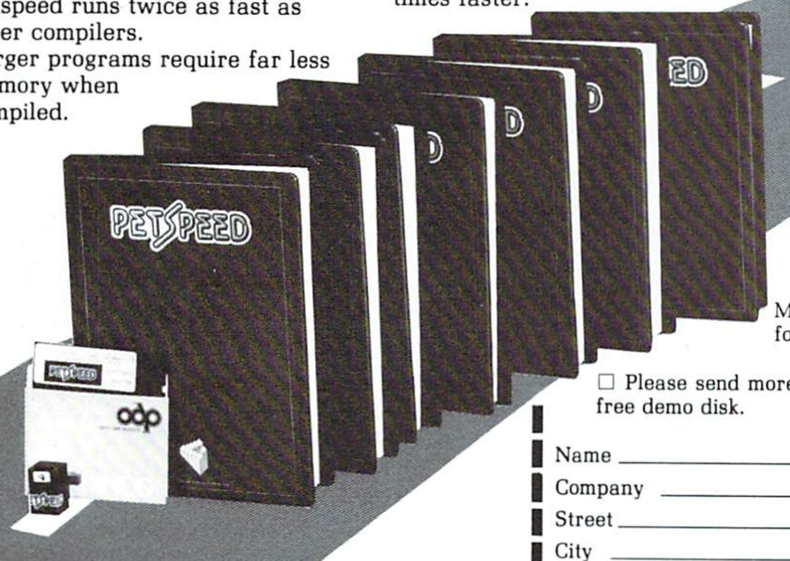
Security A security device is provided to run Petspeed, but no run-time key is necessary for compiled programs. You're free to build in your own protection. Petspeed code cannot be listed by others, so compiled programs cannot be tampered with. Your programs belong entirely to you.

Move your Commodore into the fast lane today with Petspeed. And save $25 too! Send this coupon today to

SSE

SMALL SYSTEMS ENGINEERING
222 B View Street
Mountain View, CA 94041
(415) 964-8201

Dealers: Ask for our free demo disk.
Price $325



Offer Expires 8/31/82

Save $25

Please rush Petspeed today! My disk format is ☐ 8050 ☐ 4040.
My computer is ☐ 8032 ☐ 4032.

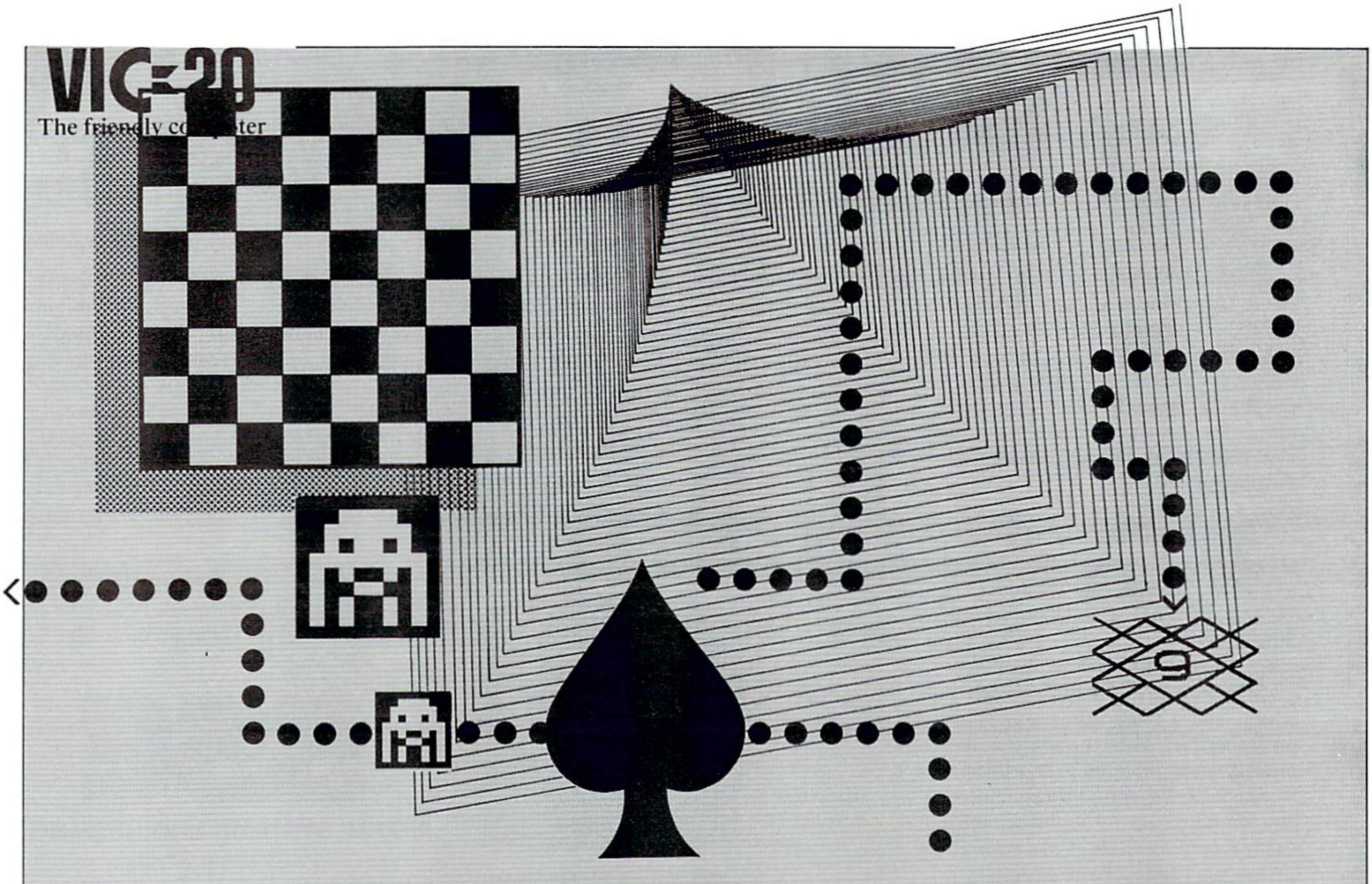
My check for $300 (plus 6½% sales tax for CA res.) is enclosed.

☐ Please send more information ☐ I'm a dealer. Send me your free demo disk.

Name ______________________
Company ______________________
Street ______________________
City ______________ State ________ Zip ________

Mail to: **Small Systems Engineering**
222 B. View St., Mountain View, CA 94041

C7



Meet the VIC 20: Part 1: Games

The VIC-20 is a great game-playing machine—most of us know that already. As an insider at Commodore I've had the opportunity to play more than 20 different VIC game cartridges already. I even hold the North American championship on a few (at least until they become available in stores).

Six Games From Japan

The first six game cartridges came from Commodore Japan, who manufactured the VIC for a year before the USA took over. These games were **Super Slot, Draw Poker, Midnight Drive, Super Alien, Jupiter Lander,** and **VIC Avenger.**

Super Slot and **Draw Poker** are based on the Las Vegas electronic gambling machines. In Slot, you bet from one to five coins, extra coins giving different lines that can pay off. The winning combinations can be displayed on the screen as well. Pulling the lever sets the one-armed-bandit in motion. This game works with a joystick or from the computer keyboard.

Draw Poker lets you play five-card-draw against the odds. You need at least a pair of Jacks to win, and can vary your bets to increase the payoff. If you win, you can bet double-or-nothing to win even more.

Midnight Drive is a much more realistic than other game machine versions (even the ones that claim to be more lifelike). Not only do you see the roadway flashing by, but you have a full instrument panel with speedometer, tachometer, four gears, engine temperature gauge, and trip odometer. The object is to go as far as possible in the time allowed. The engine stalls if you shift too fast, and overheats if you "red-line" the tachometer! The only game where adults do better than kids . . . in the beginning, at least.

Super Alien plops you into a maze along with four ugly monsters from space. Your weapon against them is the "bubble-gun," which creates traps in the maze's hallways. If you trap and destroy all the aliens, more appear in the next round! If they catch you, they eat you, so be careful.

Jupiter Lander is one of the most popular games in the group. You pilot the lander onto one of three available sites, using retro-rockets to slow your descent. I would almost swear that the gravity gets stronger as you get more points, but nobody can confirm this. My best score here is over 62,000 (I have witnesses).

VIC Avenger is the best-seller so far, and it's very similar to the Invaders games in the arcades. The action is much nicer than the cheap video game machines. In fact, even some of the subtle strategies from the arcade work on this game. Someone once taught me that if you count the number of shots you take, you can predict the value of the "mystery"

saucer that flies over the top of the screen. The 23rd shot from the start of a screen of aliens, and each 15th shot thereafter, will result in 300 points if you hit the saucer with it. The same strategy works here, too . . . I've scored over 5000 points with it.

Games to Play From the USA

Now we come to the games produced by the VIC Group in the USA. The first cartridge from here is **Radar Ratrace,** and it offers some of the best arcade action we have (along with unbelievably cute graphics and sound . . . it plays "Three Blind Mice" in the background). You control a rat in a maze, and your mission is to find and eat all 10 cheeses. You are chased by 3 enemy mice, and also have to look out for cats lurking. Since the maze is too big to see all at once, you have a radar map on the right side of the screen that shows your position and the locations of the cheeses and the enemy mice. You have a defensive weapon called (charitably) "magic stars," which you leave behind to confuse the trailing enemies. My best score here is over 85,000.

Omega Race is the first of a series of games licensed from the Bally/Midway company. With the exceptional graphics of the VIC 20, we can duplicate any video game with unmatched fidelity. Anyone who's seen Atari's PacMan for the VCS knows that their machine hasn't got the capabilities to do arcade games right. Commodore's **Omega Race** duplicates the graphics and playability of the coin-arcade game, right down to the pre-game messages. It lets you play with either joystick or paddle, which is an interesting combination.

Gorf is also on its way from Bally/Midway arcade machines, care of the USA VIC Group. Each round is a different game, as you're defending humanity against the evil Gorfian robot fleet! **Wizard of Wor** is the third game in the series, with two in the maze fighting off the Wizard's nasty "pets," and maybe even the Wizard himself!

Another game licensed from the arcade is **Pinball Supreme.** I personally think it's the best video-action game in the lot, and that goes pretty far. The sound effects are tremendous, for one thing. I never heard the VIC sound like bells before. There are dots that alternate between the letters of the word EXTRA, and they can be lit up for an extra ball. Rows of drop targets give bonuses, and lighting up the five smile faces is good for a double bonus. I've had over 200,000 on this, but I heard of someone testing it in Canada who got over 900,000.

Sargon II Chess was adapted for the VIC from bigger, more expensive personal computers like the Commodore PET and the Apple. It is considered the best computerized chess opponent available for microcomputers. This is the first availability for an under-$1000 computer. It plays either side, with adjustable difficulty levels—from fast and easy to slow and hard. I usually play against the very tricky middle levels for the right balance. I like the game's joystick feature, because it lets me lean back in my chair while I play.

A series of games for younger children is forthcoming. These games involve less complex reflexes and timing than the other video games, and can help teach the eye-hand coordination necessary to play other games.

Children's Series

Mole Attack is the first game in the series. The little moles stick their heads out of some of the holes, and you have to bash them on the head with a hammer. If you hit them on the tail end by mistake, not only do you lose points but it makes a terrible rude noise. The cutest graphics I've ever seen, too. In fact, I know a certain group of New York stock publicity people who neglect their duties for this game from time to time. My record here is 279 (whew!).

The Sky Is Falling has you using the paddle to catch objects falling from the sky. The catchers and the objects get smaller as time passes. This one's meant for the 3–6 year old who needs an interesting game requiring the simplest coordination.

Adventure Games

There is a series of adventure games that are different from the other video action games. Instead of reflexes, the most important skill is your power of deduction. The games use text rather than graphics, describing a scene for you and asking what action you want to take. The five adventures we're releasing were developed for more expensive computers by Scott Adams. They are: **Adventureland Adventure, Pirate Cove Adventure, Mission Impossible Adventure, Voodoo Castle Adventure** and **The Count.** Extra features of all adventures are the ability to stop the game at any point and store your progress on cassette tape, and a talking feature when used with the Votrax "Type'n'Talk."

In **Adventureland,** there are 13 treasures hidden throughout the land. Your goal in this introductory game is to gather all the treasures together in the right place. Can you: wake the sleeping dragon without facing her wrath? Evade the killer bees? Discover the underground caverns? Fly on the magic carpet?

Pirate Cove Adventure starts you in an apartment in London—can you find the magic word that transports you to Pirate Island? Can you then find the pirate ship? Be sure to listen to the Parrot's advice!

Mission Impossible Adventure is one of the most challenging. You are in a room with a tape recorder and manilla envelope. Play the tape, and you find your mission: to stop a nuclear power plant from exploding! You're trapped in the plant along with the mad scientist who rigged the bomb and changed the security system. Not for the faint of heart.

Voodoo Castle asks you to help free

Count Christo from a voodoo curse. Ask the medium for help, find out why there are moans coming from the fireplace, learn the secret of the hidden graveyard, and much more.

The Count places you in the castle of Count Dracula himself. You must learn how to destroy the vampire, or he'll get you! Travel the secret passages of the castle, gathering clue after clue, until you can stop Dracula.

All the games mentioned so far are available on convenient plug-in cartridges, priced from $29.95 to $39.95. All should be available by the end of the summer, if not already.

Fun & Games on Cassette

Games are also available on economical cassette tapes, selling for $59.95 for 6 tapes. The Recreation 6-Pack of tapes include 4 game tapes, along with some other entertaining programs. The games are: **VIC 21: Casino Style Blackjack, Car Chase, Slither and Super Slither** and **Blue Meanies From Outer Space.**

VIC 21 is patterned after the play in Atlantic City casinos. One or two can play against the house, with a wider range of betting options than video games, including hit, stick, double down, splitting pairs, and surrender.

Car Chase puts you in control of a car racing around a square maze-like track, running over dots to collect points. The computer controls a car coming at you in the opposite direction, which tries to crash head-on into your car.

Slither includes 2 different versions of a game where you control a slithering "worm" around the screen, trying to hit randomly appearing and disappearing boxes to collect points.

In **Blue Meanies From Outer Space,** your goal is to protect your spaceship's power plant from a rain of invading meanies. Your weapons include 2 kinds of lasers and a robot that repairs your damages.

Many more games are coming that aren't as definite as these. Commodore is in the process of locating and distributing software by outside vendors, including some game cartridges and many game tapes that should become available in the fall. We are even buying programs from VIC 20 owners. If you know of any games or other software worthy of our attention, please write me at the Commodore Consumer Products Division, so that other VIC 20 owners can enjoy the widest range of software.

—*Neil Harris*

Using Your VIC 20 to Design Useful Forms

The VIC 20 is a great game machine, but games are not its only application. You can use your VIC and its powerful graphic capabilities to design many types of forms, for both business and home. The form below, which allows the user to keep track of incoming and outgoing inventory, was printed on the VIC 1515 Graphic Printer. The blocks were created by piecing together graphic characters straight from the VIC's keyboard.

Let's talk more about modifying the program to suit your individual needs. Line 5 is probably the only line you **won't** want to change. This line directs the output to the printer. Lines 10 to 50 are the "personal" touches to the program. Line 10 prints the title of the form in enhanced letters. CHR$ (14) directs the printer to print in large boldface letters, and CHR$ (15) returns the printing to normal size. Line 18 simply prints a place to enter the date. Lines 30 and 40 print column titles. All of these names and titles can be easily changed to design your own forms.

The second half of the program prints the graphic blocks. The accompanying chart describes how to obtain the special characters from the VIC keyboard.

GRAPHIC CHARACTER	KEYS
┌	C= A
─	SHIFT *
┐	C= S
└	C= Z
┘	C= X
│	SHIFT —

These characters are used in lines 150 to 170. The size of the blocks can be changed by using different combinations of symbols. Changing the loop variable in line 100 will allow you to have more rows of blocks.

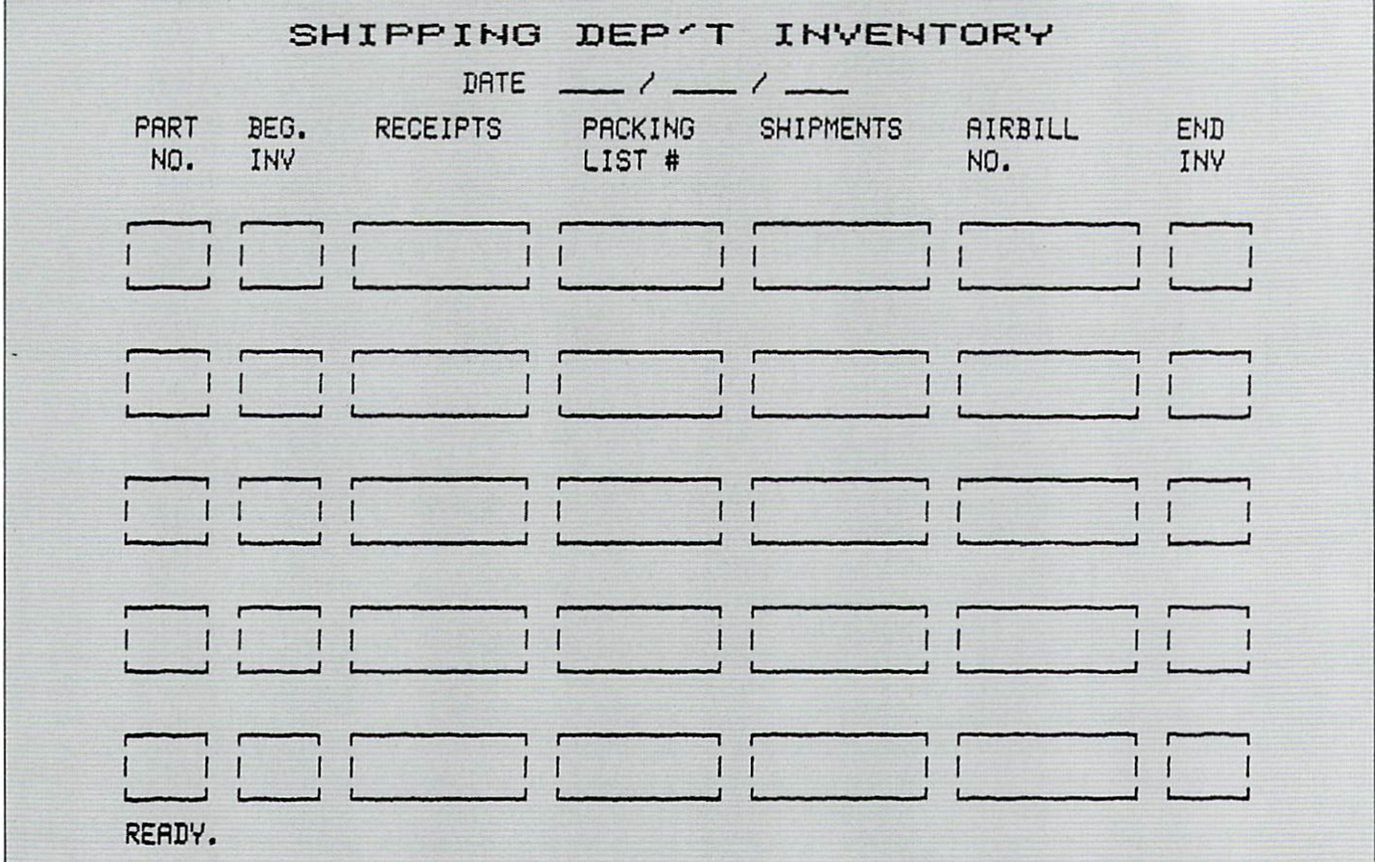
This program, with minor modifications, will allow you to keep track of just about anything, from your slide collection at home to your incoming packages at work. Next month we'll feature an enhanced version of this program that will allow you to input your data and have your VIC fill in the blocks automatically. C=

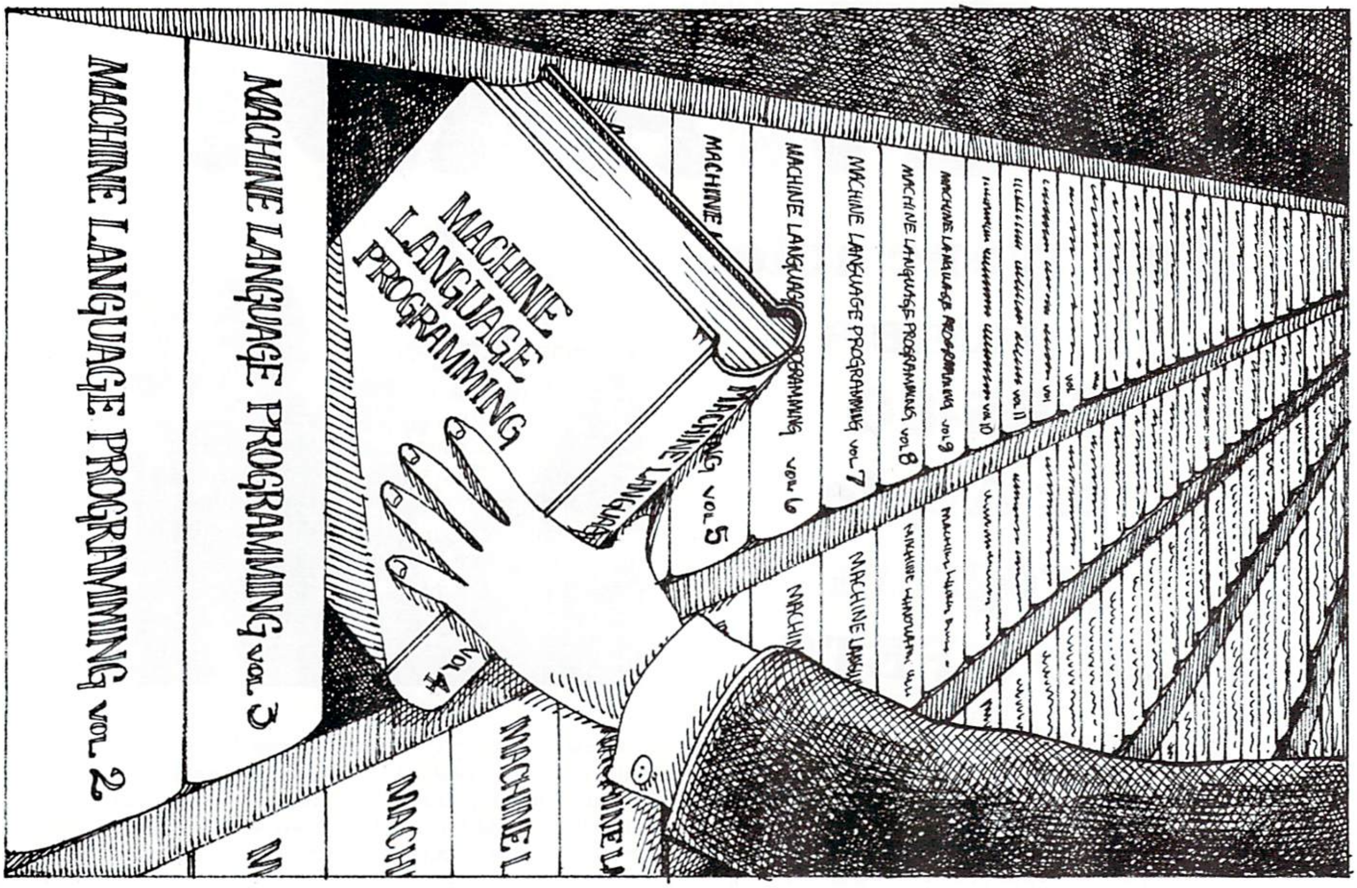
—Joseph Siciliano

```
2 REM ** VIC BUSINESS FORMS **
3 REM ** BY JOSEPH SICILIANO **
5 OPEN4,4:CMD4
10 PRINTCHR$(14)"      SHIPPING DEP'T INVENTORY"CHR$(15)
15 PRINT:PRINT
18 PRINT"                          DATE  ___ / ___ / ___"
20 PRINT:PRINT
30 PRINT"  PART  BEG.   RECEIPTS    PACKING   SHIPMENTS   AIRBILL    END
"
40 PRINT"   NO.   INV               LIST #                NO.        INV
"
50 PRINT
100 FOR A=1TO5
150 PRINT"  ┌───┐ ┌───┐ ┌───────┐ ┌───────┐ ┌───────┐ ┌───────┐ ┌─
─┐"
160 PRINT"  │   │ │   │ │       │ │       │ │       │ │       │ │
 │"
170 PRINT"  └───┘ └───┘ └───────┘ └───────┘ └───────┘ └───────┘ └─
─┘"
180 PRINT
200 NEXT A

READY.
```

VIC=20
The friendly computer





Machine Language Programming: Volume 4

The Status Quo

Now that we've been discussing the elementary facets of assembly language for a few issues, let's begin to improve the quality of our programming and "tighten" our code by using all the advantages we have available

This month, I'd like to talk about the most versatile register in the venerable 6502, the Status Register. This register is really just a set of seven flags which act as individual switches, but for our convenience the microprocessor treats all the flags as bits or a single 8-bit register.

The Status Register contains seven very useful status flags which are pictured and described below:

```
  7    6    5    4    3    2    1    0
-----------------------------------------
 N    V         B    D    I    Z    C
-----------------------------------------
```

bit 0 - C : The CARRY flag is set or unset as a result of specific arithmetic operations and can be considered the ninth bit of an arithmetic operation. It can also be modified by a set or clear carry instruction. In cases of shift or rotate instructions, the carry flag can also be used like a ninth bit. Instructions which affect the carry are: ADC, ASL, CLC, CMP, CPX, CPY, LSR, PLP, ROL, RTI, SBC, SEC.

A set carry (SEC) instruction sets the carry flag to 1. This operation should normally precede an SBC loop. This is also useful for initializing a bit to zero when using ROL. The SEC instruction affects no registers in the microprocessor and no other status flags.

The clear carry (CLC) instruction sets the carry flag to 0 and should precede an ADC loop. It is also useful for clearing a bit in memory when using the ROL instruction. Like SEC, CLC affects no other registers or flags.

bit 1 - Z : The ZERO flag is set by the microprocessor during any data movement or calculation when the 8 bit result of the operation equals zero. The flag is set when the result is 0 and unset when the result is not 0. The zero flag is not directly settable or unsettable by any instruction, but is affected by the following: ADC, AND, ASL, BIT, CMP, CPY, CPX, DEC, DEY, EOR, INC, INX, INY, LDA, LDX, LDY, LSR, ORA, PLA, PLP, ROL, RTI, SBC, TAX, TXA, TYA.

bit 2 - I : The INTERRUPT DISABLE flag is controlled by the programmer and the microprocessor to control the operations of the interrupt request (IRQ) pin. We will be discussing the use of interrupt in a future column, but basically the purpose of the interrupt disable is to disable the affects of the IRQ pin. This flag is

set by the microprocessor during reset and interrupt commands and may be unset by the CLI or PLP instructions. This bit can also be unset by a return from interrupt in which the interrupt was disabled prior to the interrupt. The interrupt flag may be set by the programmer by a SEI instruction or cleared by the CLI instruction. Instructions which affect the interrupt bit are: BRK, CLI, PLP, RTI, and SEI.

The set interrupt (SEI) instruction sets the interrupt disable to 1 and mask interrupt request during interrupt commands. This instruction affects no other registers or flags in the microprocessor.

A clear interrupt (CLI) instruction sets (unsets) the interrupt disable to 0 and allows the microprocessor to receive interrupts. This instruction also affects no other registers or flags in the microprocessor.

bit 3 - D : The DECIMAL MODE flag controls whether the adder operates as a straight binary adder or as a decimal adder for add and subtract instructions. The SED instruction sets the flag and the CLD instruction unsets it. The only instructions which affect the decimal mode flag are: CLD, PLP, RTI and SED.

The set decimal mode (SED) instruction sets the decimal mode flag to 1 and makes all subsequent ADC and SBC instructions operate as a decimal operation. This instruction affects no other registers or flags.

A clear decimal mode (CLD) instruction unsets the decimal mode flag to 0 and causes all subsequent ADC and SBC instructions to operate as simple binary operations. This instruction affects no other registers or flags.

bit 4 - B : The BREAK COMMAND flag is set only by the microprocessor and used to determine if an interrupt was caused by a BRK command or a real interrupt during an interrupt service sequence. This bit is meaningful only during the analysis of a normal interrupt sequence and can not be set or unset by any instruction.

bit 5 : This bit is called the EXPANSION bit and is not used. When analyzing the contents of the status register this bit may be set, but there is no guarantee of its state as this bit will be used in expanded versions of the microprocessor.

bit 6 - V : The OVERFLOW flag also has to do with arithmetic operation, specifically operations on signed binary numbers. When dealing with signed numbers, the 7th bit of the accumulator is called the sign field (0 = +, 1 = −), if this bit is set after an ADC or SBC loop we must correct the sign. If we are not working with signed arithmetic, we can ignore this flag. Instructions that affect the overflow flag are: ADC, BIT, CLV, PLP, RTI and SBC.

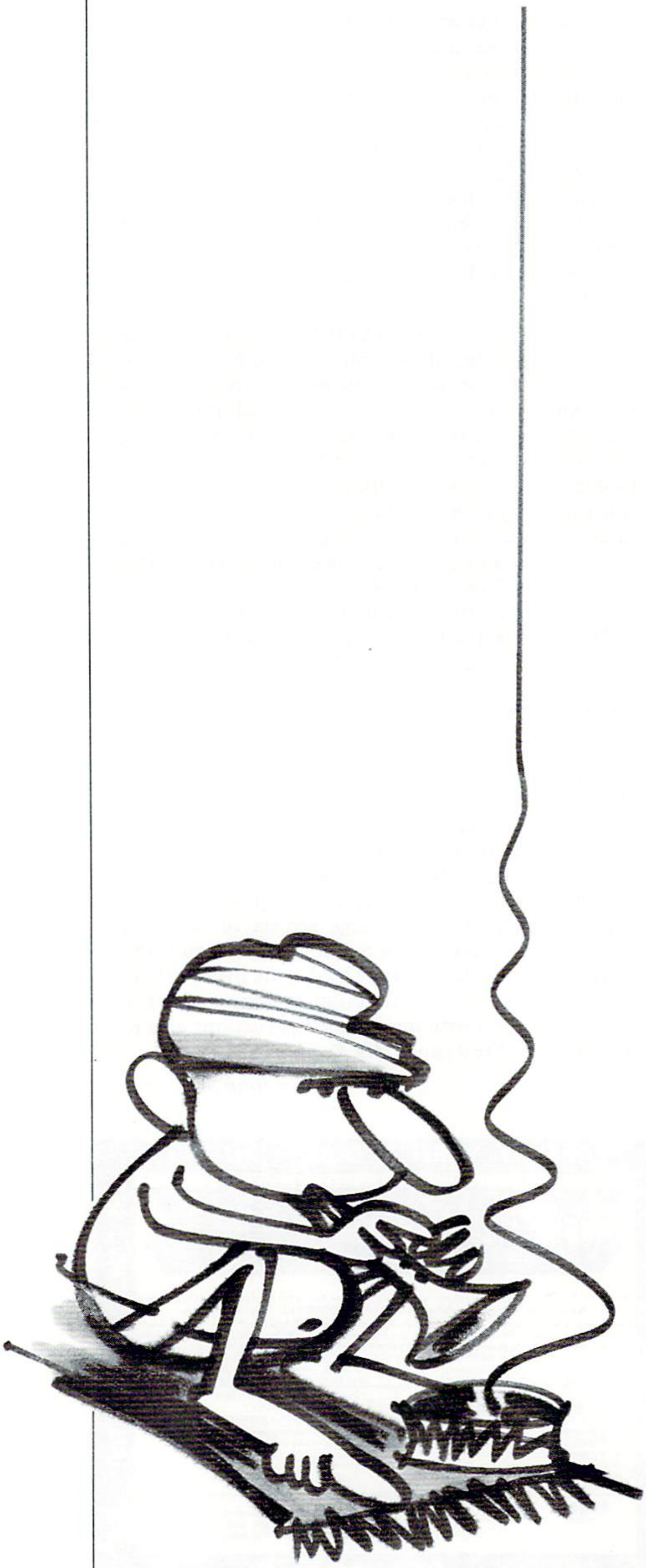
The clear overflow (CLV) instruction unsets the overflow flag to 0. This command is used in conjunction with the set overflow pin which can change the state of the overflow flag. This instruction affects no other registers or flags.

Overflow is determined, in signed arithmetic operations, when a result is not in the range of signed numbers (+127 to −128). The overflow flag will never be set when adding numbers of opposite sign because their result will never exceed that range. When adding positive numbers the microprocessor notes that bit 7 in each case is 0 and arithmetic operations yielding less than or equal to +127 the resultant bit 7 will equal 0. When adding negative numbers the bit 7 of each is recognized to be 1 and any arithmetic operation yielding less than or equal to −128 will cause the resultant bit 7 to be 1. In simplest terms, the bit 7 of each number being added determines what the resultant bit 7 must be in a non overflow situation. If the result is out of range the overflow flag is set.

bit 7 - S : This is the NEGATIVE RESULT flag, it allows the user to readily sample the sign bit (bit 7) in signed arithmetic operations. The negative result flag is set to equal bit 7 in all data movement and arithmetic. For example, after a signed add we can just test this flag to determine the sign of the result rather than trying to find some way to isolate the 7th bit. Although primarily intended as a sign bit, the negative result flag can be much very useful as an easily testable bit because most single-bit instructions, all interrupts, and most I/O status flags use bit 7 as a sense bit. Like the zero bit, this flag is not settable or controllable by the programmer and reflects the last data movement operation. Instruction which affect this flag are: ADC, AND, ASL, BIT, CMP, CPX, CPY, DEC, DEX, DEY, EOR, INC, INX, INY, LDA, LDX, LDY, LSR, ORA, PLA, PLP, ROL, SBC, TAX, TAY, TXA and TYA.

Summary:
The microprocessor treats a set of individual flip-flops as a single status register. Some of these flag bits are controllable only by the programmer, such as the decimal mode (D) flag. Others are controllable by both the programmer and the microprocessor, such as the interrupt disable (I) flag. Some flags are set or unset by almost every operation, such as the negative result (N) and zero result (Z) flags. Each flag has its own particular meaning in some specific point in time and when used in conjunction with the branch instructions give us a great amount of test capability.

—John Stockman



The Magical Vertical Line

by
Elizabeth Deal

Everybody knows that VIC is capable of magical tricks. Not to be outdone, here is a diagnostic trick for the 40-column PET/CBM. It is written on Upgrade PET (Basic 2). It should also work in 4.0 systems and possibly VIC. The information in this article is useless to 80-column people, and I don't think original ROMs can use it.

If you're still with me, the routine in lines 160–200 will answer all the questions you may possibly have about the screen . . . only kidding. And, if you have POWER (Professional Software), the code in lines 220–300 adds a command to your bag of debugging tricks, namely erasing one or more lines, this being a tiny step towards converting a 40-column PET into an 80-column PET, except I heard it can't be done.

The Problem

If you ever had trouble with the cursor not acting quite according to your wishes on some lines of the screen, while the performance of the same routine is impeccable on other lines, the problem most likely results from overprinting. You may have designed your screen output in such a way that some information is displayed first, followed by cursor home and overprinting in several remaining areas. A fine procedure, until you begin to be plagued by subtle inconsistencies.

It's my experience that most, if not all, problems are due to the wraparound of screen lines. These problems never occur when you POKE the screen and when you stay away from the edges. It's never a problem when printing sequentially. Only overprinting seems to cause trouble. And, if you've gotten a bit confused in the use of semicolons and carriage returns, things can get nasty.

A bit of digging into the wraparound handling resulted in the magical vertical line. The line turns a small nuisance into a visible test.

The difficulty with studying the screen wraparound table ($E0, decimal 224) is that the table is alive and kicking at all times. This, in turn, calls for picking up the values and storing them somehow for a further look. By the time we are through we either forget what we were looking for or the screen is gone. There are ways around it, but the line method seems to be the simplest and least destructive of evidence.

Users of POWER's backarrow instant subroutines are lucky: press shift-V and the line appears, N takes it off. Not a word is printed on the screen. If you do not have POWER you can still use the line, but you can examine only a part of the screen table at a time since you'll have to write GOSUB on a clear line and the PET, of course, insists on saying READY right on your screen. It goes without saying that GOSUB requests should be positioned away from the trouble spot.

Some Background

The way PET handles continued program lines never ceases to amaze me. It is an incredibly complex process

that doesn't fail. But what's good for a program need not be good for a fancy screen display, until we decide to capitulate to PET's rules of the game. Here are some indicators of the rules:

The wraparound table contains 25 values, one per line, initially put there by the clear-screen routine, and updated by LIST, PRINT statements and inserts. Each value is the high byte of the screen address, with the most significant bit cleared (minus 128) if the line is continued. The address of the first position on the current line (where cursor is) is in decimal 196/197, the cursor position on the line is in 198 (equivalent to POS(0)), the line number (0 to 24) is in 216. The character the PET sees under the cursor is in decimal 169. The table of low bytes of line addresses come from ROM at $E748. Finally, and this is the key to the whole story, the length of the line containing the cursor is in decimal 213, the value being 39 for single screen width lines and 79 for double screen width lines. That number is based on the wraparound table.

Query this: what would we do if Jim Butterfield hadn't written memory maps?

If you ever wondered how PET knows to refuse to insert more characters than are permitted into a double screen width line that's where it all happens. PET does not allow to extend a line past 80 positions, as signalled by the wraparound table. And PET refuses to overwrite on an already double line, hence the line is an indicator of double screen width lines. Bottom line is skipped in the process, its status known most of the time.

The obvious demonstration is on a program listing. Such programs should of course contain several double lines for illustration. Once you understand how the line behaves on a listing, you should have no trouble using the line as a diagnostic tool in printing anything on the screen where you want it.

Please note, that the line is useful only with PRINT statements. POKEing or LOADing the characters has no effect on the wraparound table. PET's messages don't count either. Combined POKE and PRINT, of course, need special handling, but you'll get the hang of it.

Incidentally, if you save screen images on a floppy, and if these screen images are BASIC program lines, you may have trouble telling your genuine program from the screen image. If you wish to know the difference, the simplest procedure is to stick in the line before saving. You can always erase it if you wish to enter a line into a program.

Leap Forward With Power

Some CBM computers allow us to press various keys to clear a line to or from the cursor. That's handy in direct mode, if you're trying to clear a line for recalculation or don't want to use DEL key which moves everything else on the screen. It is, of course, handy in modifying program lines. The erase line code in lines 220–300 isn't quite the same thing, as it only erases from the cursor, but it's enough for me. Erasing lines flows from the preceeding automatically: we can and we must carefully control whether a line is left as a single or double screen width line.

It makes sense only with POWER. Presing shifted-E erases text from the cursor to the end of the line or until the key is released, whichever comes first. The cursor returns to its original place, and you can continue typing. Pressing shifted-D deletes multiple lines, as long as the key is held down. The cursor does not return to the original place.

E is best used in BASIC text editing, D for clearing out large hunks of screen. In case of D all erased lines are released as single screen width lines, no matter what category they fell into prior to deleting. Hence you're free to use the entire area.

In case of E, things are slightly more complicated. Experiment a bit before editing real program lines. Use RETURN key with care. These are my rules:

1. If you erase all or some characters on a single screen width line (starting at position zero) then this line remains a single width line.

2. If you erase all of the double screen width line, then the line is released as two single width lines, once again, you can use it for a new program line, or for calculations, the cursor will behave correctly.

3. If you erase any part of a double screen width line, none of it will be released. The PET will continue treating it as a double line, hence you can make corrections on it, should it be a program line.

It is quite possible more rules are needed as PET's screen handling is quite elaborate. Any and all suggestions are welcome, of course.

PROGRAMMER'S TIPS

```
PROGRAM:   MAGIC LINE*

110 REM"V←160
120 REM"H←170
130 REM"E←220
140 REM"D←230
150 REM---VLINE/NO VLINE-----------------
160 L$=CHR$(165):GOTO 180
170 L$=CHR$(32)
180 LN=216:PRINT CHR$(19);:FOR J=0 TO 21
190 PRINT L$:IF PEEK(LN)<24 THEN NEXT J
200 RETURN
210 REM---ERASE/DELETE-------------------
220 E8=0:GOTO 240
230 E8=1
240 E1=213:E3=224:E4=151
250 E7=PEEK(216)+E3:E6=POS(0)
260 E5=PEEK(196)+256*PEEK(197)
270 FOR J=E6 TO PEEK(E1):IF PEEK(E4)<>255 THEN POKE E5+J,32:NEXT J
280 IF E6+J-00=0 THEN FOR J=0 TO 1:POKE(E7+J),PEEK(E7+J)OR 128:NEXT J
290 IF E8 THEN IF PEEK(E4)<>255 THEN PRINT:GOTO 250
300 RETURN
310 REM----------------------------------
```

Features of the 6845 Video Controller

by
Jim Holtom
Control Microsystems

All Commodore machines that have 12″ monitors (8032, fat 4032, 8096, and SuperPET) employ a device known as the 6845 video controller to generate the video signals. Some come with a 6545, which is the same as the 6845.

The chip has 18 programmable registers that are accessed through 2 memory locations at $E880 and $E881 (59520 and 59521). To read or write a register, the number of the register (0–17) is stored in 59520 and that register can then be accessed through 59521. This method saves considerably on address space.

The device is used to control such functions as character height, cursor size, horizontal position, vertical resolution, etc. Here is a summary of the registers and their functions:

R0—Horizontal Total Register. Horizontal frequency equalling the total of displayed plus non-displayed "character time units" minus 1.

R1—Horizontal Displayed Register. Number of displayed characters per horizontal line.

R2—Horizontal Sync Position. Controls horizontal positioning.

R3—Horizontal Sync Width. Four bits which control the width of the horizontal sync pulse.

R4—Vertical Total. The vertical frequency is controlled by R4 and R5.

R6—Vertical Displayed. Number of displayed character rows on the video.

R7—Vertical Sync Position. Controls vertical positioning.

R8—Interlace Mode. 2 bits which determine whether interlaced or non-interlaced mode is employed.

R9—Maximum Scan Line Address. Five bits determining the number of scan lines per character row including spaces.

R10—Cursor Start. Seven bits for cursor start scan line and blink rate.

R11—Cursor End. Five bits for cursor end scan line.

R12—Start Address. R12 and R13 control the first address put out as a refresh address after vertical blanking. R12 is the low 8 bits and R13 is the hi 6 bits of the address.

R14—Cursor Register. This 14 bit register stores the cursor location. R14 is the 8 low and R15 is the 6 hi bits.

R16—Light Pen. Fourteen bits of R16 and R17 store the contents of the Address Register when the light pen strobe goes high.

All registers are write only with the exception of R14, R15, R16 and R17. This means the registers 0–13 can only be POKEd. PEEKing these registers will return invalid results.

Interesting Effects

In the early days, it was possible to blank the screen of the 2001 PET. This was useful for visual effects. BASIC 2 machines omitted this capability which was unfortunate. Now it's possible again! The following program demonstrates screen blanking for 12″ monitor machines:

```
10 FOR J = 1 TO 2000 : PRINT "*"; : NEXT :
   REM FILL SCRN
20 POKE 59520, 1 : REM HORIZ DISPLAY REG
30 POKE 59521, 0 : REM NO WIDTH =
   NO DISPLAY
40 FOR J = 1 TO 2000 : NEXT : REM DELAY
50 POKE 59521, 40 : REM VIDEO BACK ON
```

Notice that the register offset (59520) need not be set again to access the same register as before. Default display width for 40 and 80 columns screens is "40".

The horizontal and vertical positioning of the display area can be altered by POKEing different values into R2 and R7. R2 default is 41 and R7 default is 29.

The number of displayable character columns may be modified by POKEing different values into R1. R1 default is 40. Likewise, the number of displayable character rows can be changed using R6. R6 default is 25.

The machine makes use of the controller for things like windowing, and text/graphic modes. Several other effects could certainly be achieved for use in games and other applications. The combinations and permutations are virtually endless!

NOTE:

A word of caution. Uneducated experimentation with the 6845/6545 can potentially crash the computer. The effects involving the registers discussed are all safe to play with but other registers should be left alone unless you know what you're doing.

What You See Is What You Get

by
Elizabeth Deal

We can learn little things about how machine language functions by watching it actually happen. Tiny exercises can reveal things that usually appear hidden from sight. We will use a bit of the screen as if it were the only memory in the PET.

The reason we can do it is that to the PET, the screen is just an image of a hunk of memory. The 1000-byte screen memory boundries are 32768 and 33767, or, in hexadecimal, $8000 and $83E7, the latter being 24 bytes short of $8400. We will use part of the area for our work and turn most of it over to the PET's machine language monitor. It will be a bit tight, so be careful to type things in correctly in order to fit everything where it should.

As we do the work, things may go wrong. PET may crash—it may not respond to you at all if you type things incorrectly. Don't worry about it. The amount of writing is insignificant; retyping won't be a problem. If you have Supermon or Extramon, load it and enable it. If you have Power, turn it on; its repeating cursor key will be handy. If not, use any repeating key routine. But all you really need is your PET. The other things are just handy aids.

Now clear the screen and type SYS4 ⟨return⟩ to enter the machine language monitor. When the cursor lands on a dot, ask the PET to display contents of 48 bytes of its screen memory, by typing

```
.M 8370 839F <return>
```

The monitor will answer back with six lines of eight bytes each. All contents should be $20, meaning spaces. Let's experiment with what is there.

1. Move the cursor up to the first display line and overtype several $20's with another value ranging from $00 to $FF. Pick any place, but push RETURN when you're done with a line, so the PET can hear you. What happens? Did you put a number in $8370? How about $8388, $8397 and $839F? Where are they on the bottom display lines? What are they? Change the values.

2. Can you compare the screen code to the ASCII code you use in BASIC? Is there a pattern? Do all values show up? Why or why not?

3. Carefully take the cursor down to the rightmost part of the first display line and using DEL key wipe it all out. Then bring the cursor up to the .M line and push RETURN to get the memory display. If you've done things right, all values should be $20. Are they? If not, why not?

4. Enter several values in the top part and press RETURN. Take the cursor down and on the display line enter some characters. Now bring the cursor up and do as in point 3—ask the PET for memory display after the change. What do you see? Why?

5. If you don't use a freely available (public domain) Supermon or Extramon, read the following to see two of the many commands you're missing, but skip to exercises in point 9.

6. Right under the six display lines type carefully without extra spaces and without any errors:

```
.F 8370 8397 20 <return>
```

That's a memory fill instruction. Beats typing it by hand, doesn't it? Now try:

```
.F 8370 8378 35 <return>
```

What do you see?

7. Let's move what you see a bit to the right, by typing correctly:

```
.T 8370 8378 8380 <return>
```

This is memory block-move command in action. It transfers values from one place to another, leaving the original values intact. It's a smart command, it will not destroy code, as you'll see in this:

```
.T 8380 8388 8382 <return>
```

Note the overlap. Try the T instruction to reverse the process.

8. Let's see what exists somewhere else in the PET, for instance on the first line of the screen:

```
.T 8000 8027 8370 <return>
```

Can you make sense out of it? Let's see what is in the registers, timers, and ports in the 6522 VIA

```
.T E840 E84F 8370 <return>
```

Bear in mind that as soon as you saw the result, many values in the VIA have changed. You can confirm this is so by doing the T command again: move the cursor up to T and hit RETURN. Try peeking at some other locations in PET's memory.

9. We can do something else. We'll write a tiny visible program to operate on visible data. Clear the screen. You're still in the Monitor.

10. Display the same hunk of memory as before:

```
.M 8370 839F <return>
```

Fill bytes $8388 to 838F with other values and RETURN. Can you see the data below?

11. Enter a program to move the characters four positions to the right, as in T command above, but also to fill the original positions with blanks. Type these lines over the existing display without any errors, hitting return after each eight bytes. Don't enter the dots. They are there to visually separate instructions.

```
.:  8370 A0 20.A2 03.BD 88 83.9D
.:  8378 90 83.98.9D 88 83.CA.10
.:  8380 F3.00.00 00 00 00 00 00
```

Did you see this code being entered into memory as you pushed RETURN?

The code we wrote would look like this in assembly language:

```
     LDY #$20
     LDX #$03
LOOP LDA $8388,X: STA $8390,X
     TYA
     STA $8388,X
     DEX: BPL LOOP
     BRK
```

The dots, above, separate those instructions. You can try to match the machine code instructions you typed in to this code by looking at the numbers. 'LOOP' is a label (a goto-type target address), it has no instruction. BPL LOOP translation may not look obvious when it is coded 10 F3. The meaning of F3 is for the PET to go back from $FF to $F3 where $FF points to the second byte in the 10 F3 pair. If you count backwards, you'll see that F3 lands on the 9D instruction.

The same code would, more or less, look like this, in BASIC:

```
10 Y=32: X=3: M1=33680: M2=M1+4
20 FOR I=X TO 0 STEP-1
30 A=PEEK(M1+I)
40 POKE(M2+I),A: POKE(M1+I),Y
50 NEXTI: STOP
```

12. Verify typing. The code is little and can be reentered, but why waste time? Note that the code we just placed on the screen is volatile. If you clear the screen or overtype, it will vanish. Check its existence once in a while and if in doubt, press RETURN several times starting at line 8370 to reenter the code.

13. Move the cursor to just below the six-line display and type

```
.R <return>
```

This shows the current status of the registers. The first position is the program counter (PC). It contains some value. Move the cursor to the value and overtype 8370. This places the address of the first instruction of the code you just entered into the Program Counter. To execute the code type

```
.G <return>
```

G stands for "go there." If you've crashed, reset the PET and repeat from step 9. Otherwise, check the results. Did the four characters move to the right? Are original positions blank? If not, go back to step 12. Keep looping until it works.

14. You may have noticed some additional information PET gave you near the bottom. '*B' tells us the PET hit a BRK ($00) and the program counter points to the next executable location.

15. Modify the program. Try changing the value of Y in $8371 from $20 to some other number. Run it by doing point 13. What do you see? Change it back. What does Y register do? What does X register do?

16. Change value of X in $8373, but keep it under 8, we don't have room to move big things. What do you see? Why? Display our work area memory by hitting RETURN on the three lines beginning with $8388. Change X back to $03.

17. Change 'branch on plus' (BPL) in $837F to 'branch on not zero' (BNE): overtype $10 with $D0. Watch carefully and explain what happened. Change it back.

18. Eliminate TYA: STA $8388,X by overtyping 'no-operation' or $EA in locations $837A to $837D. Execute the code. What happened? Put the original code back in.

19. Change the destination address from $8390 to some other number. Stay within the $8388–$839F range, though, to avoid trouble. What happens? You may change other addresses. While doing so coordinate it with your data that was originally placed in line $8388.

21. Invent your own things. Abandon the strict observance of the memory limits I recommended earlier for convenience (but stay within the screen area!). See where you can or cannot put things. If you've gone through the whole process you won't have much trouble fixing errors. At worse the PET will crash. It's worth learning why it crashed by crashing while the typing investment is small.

22. If you have Power, in the future you can write yourself some instant key macros for the up and down travel of the cursor from the top to the bottom display. Try it.

These experiments don't do anything and are insignificant in themselves. But they should make you more comfortable in use of the screen editor in the Monitor mode, and might even make the Monitor feel as friendly as PET's BASIC.

I actually use the screen as a debugging arena whenever possible. Throwing results tightly on the screen makes coding errors immediately apparent, whereas asking for a memory display is tedious, uses up half the screen and still requires too much thinking. C=

BASIC Plotter

by
Paul Higginbottom
Commodore Canada

This program will plot random lines using the "quarter-square" graphics characters. Although it's a program in itself, it could easily be made into a subroutine.

The program has been set up for 80 column screens (line 9040). Notice "LL" (Line Length) is multiplied by 2 in lines 2020 & 2030? Since the quarter squares use up half a character space in the "x" direction, an 80 column screen can have up to 160 "half-characters" horizontally. Similarly, on 25 lines there can be up to 50 half characters vertically ("y" direction). For 40 column screens you'll need to change LL to 40; the second parameter remains the same since both have 25 lines.

Line 2000 clears the window (if one set), the screen, and sets graphics mode (no gap between lines). If you like, substitute CHR$(142) with 'esc-rvs-N' and stick it inside the quotes.

```
2000 PRINT"[HM HM CLR]"CHR$(142)
2010 GOSUB 9000
2020 X1=INT(RND(TI)*LL*2) :
  Y1=INT(RND(TI)*50)
2030 X2=INT(RND(TI)*LL*2) :
  Y2=INT(RND(TI)*50)
2040 GOSUB 3000 : Y1=Y2 : X1=X2 : GOTO 2030
3000 REM ********** PLOT A LINE **********
3010 DX=X2-X1 : DY=Y2-Y1 : X=X1 : Y=Y1
3020 L=SQR(DX*DX+DY*DY) : IF L=0 THEN
  3040
3030 XI=DX/L : YI=DY/L
3040 GOSUB 8000 : IF (ABS(X2-X)<=ABS(XI))
  AND (ABS(Y2-Y)<=ABS(YI)) THEN RETURN
3050 X=X+XI : Y=Y+YI : GOTO 3040
8000 REM ********** PLOT X, Y **********
8010 TX=INT(X+IR):TY=INT(Y+IR)
  :SQ=AM(TX AND AM, TY AND AM)
8020 P=BS+TX/DV-INT(TY/DV)*LL : POKE P,
  C(I(PEEK(P))OR SQ) : RETURN
9000 REM ********** SETUP **********
9010 DIM C(15), I(255), AM(1,1)
9020 FOR I=0 TO 15 : READ C(I) : I(C(I))=I : NEXT
9030 FOR I=0TO1 : FOR J=0TO1 :
  AM(J,I)=(J+1)*4↑I : NEXTJ,I
9040 LL=80 : BS=32768+24*LL : DV=2 : AM=1 :
  IR=.5
9050 DATA 32, 123, 108, 98, 126, 97, 127, 252, 124,
  255, 225, 254, 226, 236, 251, 160
9060 RETURN
```

The subroutine at 9000 sets up an array with the 16 possible combinations of the quarter squares. BS is the base address or the POKE address of the bottom left corner of the screen.

All plotting efforts are performed by the two subroutines at 3000 & 8000. Subroutine 3000 plots a line from x1,y1 to x2,y2 by plotting several points (sub 8000). At the same time, subroutine 8000 must determine if there is already a point in a character space. If there is, the POKE information must not interfere with existing points. Lines 200X are used for plot criteria generation. The above merely plots random lines. For something more meaningful, try substituting with these:

```
2020 X1=0 : Y1=1
2025 FOR X2=0 TO 159
2030 Y2=EXP (X2/31.4)
2040 GOSUB 3000 : Y1=Y2 : X1=X2 : NEXT : END
```

```
2020 N=6 : C=3.1415926/160 : X1=0 : Y1=25
2025 FOR X2=0 TO 159
2030 Y2=25 + 24 * SIN(X2 * N * C)
2040 GOSUB 3000 : Y1=Y2 : X1=X2 : NEXT : END
```

```
2020 N=8 : C=3.1415926/160 : X1=0 : Y1=50 :
  DC=100
2025 FOR X2=0 TO 159
2030 Y2=25 + 24 * COS(X2 * N * C) * EXP(-X/DC)
2040 GOSUB 3000 : Y1=Y2 : X1=X2 : NEXT : END
```

The first plots an exponential curve. Notice the Y origin is set to 1 rather than 0. This accounts for a slight inaccuracy as the plotter draws horizontal lines using the top "half-character" rather than the bottom half-character. This could be changed by modifying the character table at 9050.

The second draws a SINE curve starting half way up the screen (Y1=25). The variable N represents the number of half cycles displayed (N=6 will draw 3 complete cycles).

The last one is a decaying COSINE wave, origin at top-left (Y1=50). For higher decay rates, use lower values in DC.

Finally, with little effort you could use the plotter routine to draw axes for your functions.

April Addendum

In the April/May issue, we published an article entitled "An EASY Cursor Positioning Routine" (page 82). The following two brief programs, which were omitted from the article, will assist you in moving the cursor to an X,Y location on the screen. The first program is a BASIC poker that readjusts the top of memory in order to secure the routing and pokes in the appropriate code for the routine. Also provided is a loader program that first readjusts the top of memory pointers and the start of variable pointers, then loads in the program version of the routine, and finally loads in your first application program.

```
 10 REM *****************************************
 20 REM *****   CURSOR ROUTINE POKER        *****
 30 REM *****************************************
 40 :
 50 POKE 53,126 :REM **** RESET TOP OF MEMORY ****
 60 :
 70 SA=32256:EA=32256+256
 80 FOR I = SA TO EA:READ V:POKE I,V:NEXT I
 90 :
 100 DATA 32,19,126,32,47,126,32,57,126,32,67,126,32,36,126
 110 DATA 32,112,0,96,165,127,141,246,126,165,128,141,247,126,169
 120 DATA 234,133,127,133,128,96,173,246,126,133,127,173,247,126,133
 130 DATA 128,96,32,184,126,32,105,126,32,123,126,96,32,184,126
 140 DATA 32,144,126,32,162,126,96,32,118,0,201,44,240,3,76
 150 DATA 178,126,32,112,0,201,34,240,3,76,178,126,32,112,0
 160 DATA 201,34,240,10,201,0,240,80,32,210,255,76,87,126,96
 170 DATA 240,71,176,69,32,195,126,173,245,126,201,25,176,59,141
 180 DATA 243,126,96,174,243,126,169,19,32,210,255,224,0,240,8
 190 DATA 169,17,32,210,255,202,208,250,96,240,32,176,30,32,195
 200 DATA 126,173,245,126,201,80,176,20,141,244,126,96,174,244,126
 210 DATA 224,0,240,8,169,29,32,210,255,202,208,250,96,32,36
 220 DATA 126,76,0,191,169,44,160,0,209,119,208,242,76,112,0
 230 DATA 56,233,48,141,245,126,32,112,0,144,7,201,44,240,32
 240 DATA 76,178,126,56,233,48,72,169,0,174,245,126,24,105,10
 250 DATA 202,208,250,141,245,126,104,24,109,245,126,141,245,126,32
 260 DATA 112,0,96,0,0,0,0,0,170,170,170,170,170,170,170
 270 DATA 170,170
 280 NEW
READY.
```

```
 10 REM **********************************************
 20 REM ***** RESET TOP OF MEMORY SO CURSOR ROUTINE *****
 30 REM ***** IS SAFE AND RESET START OF VARIABLES  *****
 40 REM ***** SO THAT MENU PROGRAM FITS.            *****
 50 REM **********************************************
 60 :
 70 POKE 53,126:POKE42,162:POKE43,5
 80 :
 90 IF L=1 THEN GOTO 200:REM **** IF CURSOR LOADED ****
 100 :
 110 L=1:DLOAD "CURSOR"
 120 :
 200 DLOAD "MENU TEST"
```

What's the Key?

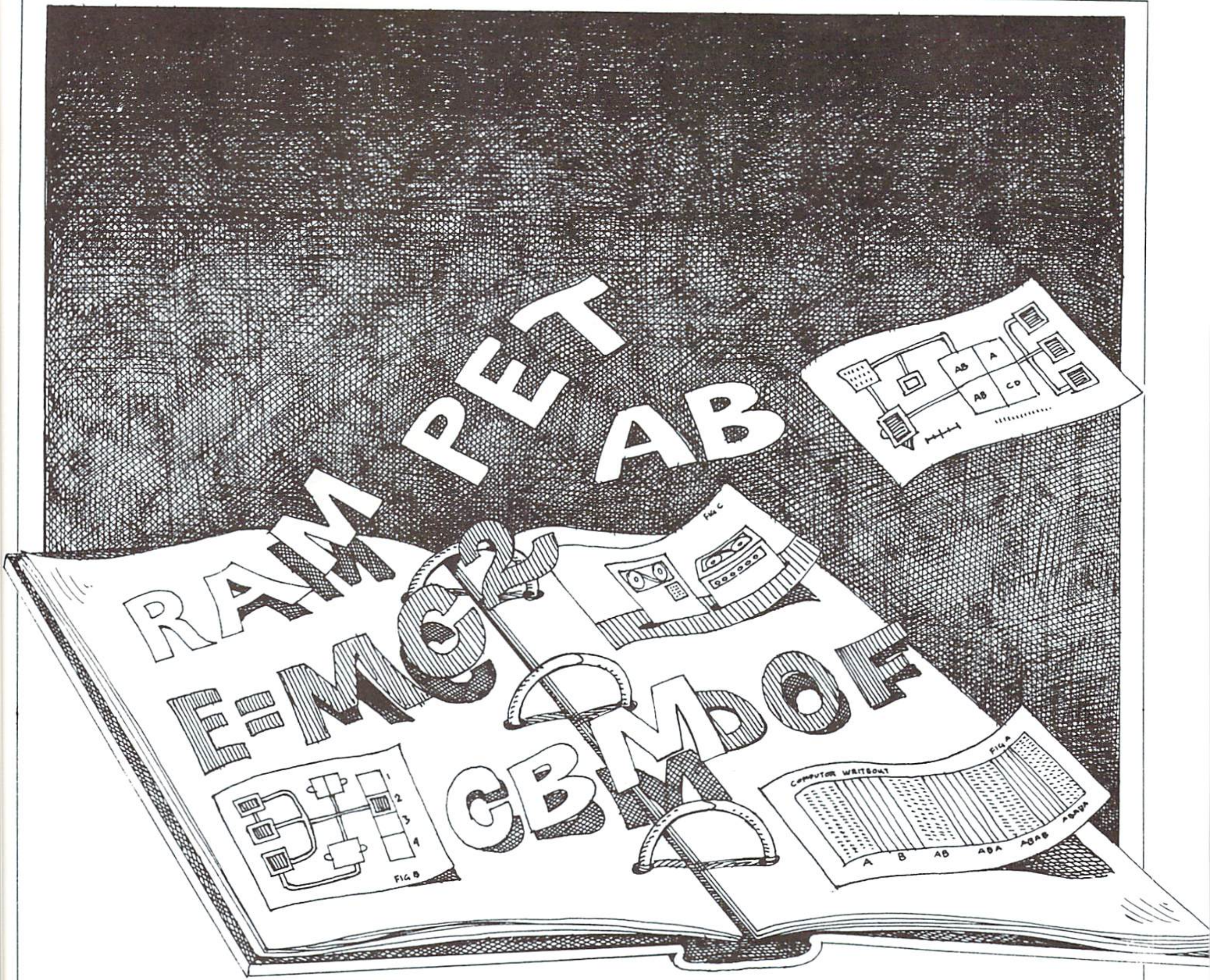
Several requests have recently been made to clarify the differences in various PET/CBM keyboards. One of the best references we can provide is an article on that very subject in the October issue. Rather than send you scurrying for that specific issue, we've decided to reprint a comprehensive list of keyboard differences. Read carefully.

KEY#	PET-GR-ORIG/UPGR	KEY#	CBM-40COL B4	KEY#	CBM-80COL B4
1	=	1		1	
2	.	2		2	
3		3	COLON AND *	3	STOP AND RUN
4	STP AND RUN	4	STOP AND RUN	4	
5	<	5	9 AND)	5	
6	SPC	6	6 AND &	6	
7	[	7	3 AND #	7	
8	RVS AND OFF-RVS	8	←	8	
9	-	9	1	9	TAB
10	0	10	/ AND ?	10	
11	R-S	11		11	
12	>	12	CLS AND HOME	12	
13	SPC	13	M	13	RET
14	]	14	SPACE	14	
15	@	15	X	15	
16	L-S	16	RVS AND OFF-RVS	16	
17	+	17	2	17	CD AND CU
18	2	18		18	RVS AND OFF-RVS
19		19		19	CLS AND HOME
20	?	20	0	20	DEL AND INST
21	COMMA	21	COMMA AND <	21	
22	N	22	N	22	
23	V	23	V	23	
24	X	24	Z	24	
25	3	25	3	25	
26	1	26		26	
27	RET	27		27	
28	SEMIC	28	.	28	
29	M	29	. AND >	29	CR AND CL
30	B	30	B	30	
31	C	31	C	31	
32	Z	32		32	SPACE
33	*	33	4	33	
34	5	34	[	34	
35		35	O	35	
36	COLON	36	CD AND CU	36	
37	K	37	U	37	
38	H	38	T	38	
39	F	39	E	39	
40	S	40	Q	40	
41	6	41	DEL AND INST	41	
42	4	42	P	42	
43	RET	43	I	43	
44	L	44	\	44	COMMA AND <
45	J	45	Y	45	- AND =
46	G	46	R	46	. AND >
47	D	47	W	47	/ AND ?
48	A	48	TAB	48	

```
49 /
50 8
51
52 P
53 I
54 Y
55 R
56 W
57 9
58 7
59 ↑  AND π
60 O
61 U
62 T
63 E
64 Q
65 DEL AND INS
66 CD AND CU
67
68 >
69 \
70 '
71 $
72 QUOTE
73 CR AND CL
74 CLS AND HOME
75 ←
76 <
77 &
78 %
79 #
80 !
81
82
83
84
85
86
87
88
89
90
155
174
176
177
178
179
180
181
182
183
184
185
192
219
220
221
222
223
```

```
49 6
50 @
51 L
52 RET
53 J
54 G
55 D
56 A
57 5
58 SEMIC AND +
59 K
60 ]
61 H
62 F
63 S
64 ESC
65 9
66
67 ↑
68 7
69 0 IN TOP ROW
70 7 AND '
71 4 AND $
72 1 AND !
73
74
75 CR AND CL
76 8
77 - AND =
78 8 AND <
79 5 AND %
80 2 AND QUOTE
81
82
83
84
85
86
87
88
89
90
155
174
176
177
178
179
180
181
182
183
184
185
192
219
220
221
222
223
```

```
49 1 AND !
50 2 AND QUOTE
51 3 AND #
52 4 AND $
53 5 AND %
54 6 AND &
55 7 AND '
56 8 AND <
57 9 AND >
58 COLON AND *
59 SEMIC AND +
60
61
62
63
64
65 A
66 B
67 C
68 D
69 E
70 F
71 G
72 H
73 I
74 J
75 K
76 L
77 M
78 N
79 O
80 P
81 Q
82 R
83 S
84 T
85 U
86 V
87 W
88 X
89 Y
90 Z
155 ESC
174 .
176 0
177 1
178 2
179 3
180 4
181 5
182 6
183 7
184 8
185 9
192 @
219 [
220 \
221 ]
222 ↑
223 ←
```



Excerpts from a Technical Notebook

Number Juggling

Formatting numbers on the screen can cause problems when the TAB function is used. If numbers are to be printed in columns, then it would be nice to ensure that the decimal point of the numbers always line up in the column.

For example, if the number is simply TABbed onto the screen, and the number is a "0", then it will appear at the left hand side of the column, which doesn't look very smart. It is therefore necessary to use the length of the number (ie. the number of digits including decimal points) to drive the TAB expression. A number has a leading space and a trailing cursor right which needs to be taken into consideration. The LEN function counts the number of characters in a string. In order to use LEN it is first necessary to convert the number into a string using STR$. The number of characters in the number is given by:

X = LEN (STR$ (A)) − 1

. . . where A is the number. The trailing cursor right is ignored by STR$ but 1 is subtracted to take account of the leading space. So now, taking the above example, if we were to TAB (10-X) we would be in business, right? No, not quite yet.

There are a couple more possibilities that should be considered. Sometimes it is desirable to tack leading zeroes onto integer numbers (ie. to display "0038" rather than "38"). With decimal numbers, we may want a specified number of digits following the decimal place, followed by trailing zeroes which the PET does not do for us. All of this will affect the positioning of numbers output to the screen.

Without question, the easiest way to handle number formatting is to first turn the number into a string:

```
A$ = MID$ ( STR$ (A) , 2 )
```

STR$ converts "A" to a string and the MID$ function is used to take the 2nd character onwards, thus removing the leading space. To add leading zeroes we use the RIGHT$ function:

```
A$ = RIGHT$ ("000000" + A$, 4 )
```

Assuming "A" is an integer, the above will produce a 4-character string. The number of leading zeroes will be dependent on the size of "A" (i.e. "16" produces "0016" and "1024" stays "1024"). For compactness, all of the above could have been done with:

```
A$ = RIGHT$ ("000000" + MID$
   ( STR$ (A),2) , 4)
```

Rounding decimal numbers and adding trailing zeroes is a little trickier. First we must decide how many significant digits are to follow the decimal place. The following is an example for 2 significant digit truncating:

```
A = 1035.55534
A = INT (A * 100) / 100
PRINT A
   1035.55
```

The above merely moves the number two places to the left, chops off the fractional part, and then moves it back two places to the right. However, this is not rounding but rather truncating, which is not the same. For rounding, we must first decide what degree of rounding is desired. Most often, numbers are rounded to 2 decimal places, or "to the penny". Our example then becomes, simply:

```
A = 1035.55534
A = INT (A * 100 + .5) / 100
PRINT A
   1035.56
```

The same result would be accomplished by first adding .005 to A and multiplying that by 100, but BASIC is more accurate at decimal arithmetic that lies closer to the decimal point.

Numbers that end up with 1 decimal or less will need trailing zeroes. Once again, this can be done with a string manipulation. For example:

```
A = 1035.59534
A = INT ( ABS(A) * 100 + .5) / 100
PRINT A
   1035.6
S$ = CHR$ (32 - (V<0) * 13)
A$ = MID$ ( STR$ (INT(A)) ,2)
DP = INT ((A - INT(A)) * 100 + .5)
A$ = A$ + "." + RIGHT$ ("00" + MID$
  (STR$(DP) ,2) ,2)
A$ = RIGHT$ ("    " + S$ + A$ ,10)
PRINT A$
   1035.60
```

This probably looks sort of clumsy but it's designed to do everything; trailing zeroes; positive and negative rounding; and decimal point aligning. Notice that we've taken the ABSolute value of A before entering the routine. S$ will be either a space or a minus sign. Then we grab the INTeger part of A into A$. DP is used to take the decimal part of A, and round it on the left of the decimal point. Next we build it all together by taking the integer part, adding our own decimal point, followed by the decimal part. As usual, the leading space is stripped off DP with MID$. "00" is added to this (remember, DP could have a value of say 04 or an even 0) and RIGHT$ comma 2 gives us our 2 decimal places. Lastly we add some leading spaces (which could also be zeroes) and then we stick the sign S$ of the front.

Easy, right? Well, you might have done it differently but this was a string juggling exercise for the practice.

Variable Flip-Flop

Programs can be shortened a great deal with a little thought and an active imagination. For example, it is often necessary to set a flag if a condition is met or to compliment the flag. One might code:

```
1200 IF FLAG = 0 THEN FLAG = 1 : GOTO 1220
1210 FLAG = 0
1220 . . .
```

On consideration, the statement

```
1200 FLAG = 1 - FLAG
1210 . . .
```

will be seen to have the same effect.

Screen Codes To ASCII

This program is a screen dump routine which makes it possible to copy the contents of the screen onto the printer. This subroutine has been presented several times but we'll be looking at technique as opposed to operation.

```
5000 OPEN 4, 4
5010 FOR J = 0 TO 999
5020 P = PEEK (32768 + J)
5030 GOSUB 5500
5040 IF P< 64 THEN P=P+ 64 : GOTO 5090
5050 IF P<126 THEN P=P+128 : GOTO 5090
5060 IF P<128 THEN P=P+ 64 : GOTO 5090
5070 IF P<191 THEN P=P- 64 : GOTO 5090
5080 IF P=255 THEN P = 191 : GOTO 5090
5090 PRINT #4, CHR$ (P);
5100 X=X+1 : IF X=40 THEN PRINT#4 : X=0 :
  F=0
5110 NEXT
5120 CLOSE 4
5130 RETURN
5500 REM REVERSE FLAG
5510 IF F=1 AND P>127 THEN 5550
5520 IF P>127 THEN F=1 : PRINT#4,"[RVS]"; :
  GOTO 5550
5530 IF F=0 AND P<127 THEN
5540 IF P<127 THEN F=0 :
  PRINT#4,"[RVS OFF]";:GOTO5550
5550 RETURN
```

As you can see, the routine begins PEEKing the screen into P. The subroutine at 5500 deals with reverse field characters. Then P is converted to its corresponding CHR$ value and it's sent to the printer. Lines 5010 and 5100 are set up for 40 columns but this can easily be changed to 80.

Now let's have a closer look at those five nasty IF statements. If we look at the differences between PEEK/POKE codes and ASCII, it becomes apparent that only the 3 most significant bits (bits 5 [32], 6 [64], and 7 [128]) are changed. "Aha, a bit of boolean algebra will solve this problem!" Using the OR and AND functions, it is possible to make the conversion with just one line! Thus if P=PEEK(32768), the top left corner of the screen, then:

```
C = (P AND 127) OR ((P AND 64)*2) OR ((64-P
AND 32)*2)
```

where C is the corresponding ASCII character. Here's the new program:

```
5000 OPEN 4, 4 : T=40 : S=32768
5010 FOR J = S TO 33767 : P=PEEK(J)
5020 R$ = CHR$(146-(PAND128))
5030 R$=LEFT$(R$,-(P>127ANDPEEK(J-1-
  (J=S))<1280RP<128AND PEEK
  (J-1-(J=S))>127))
5040 P=(P AND 127)OR((P AND 64)*2)OR((64-P
  AND 32)*2)-((PAND127)=0)*64
5050 PRINT#4, R$;CHR$ (P);
5060 IF (J-32807)/T = INT((J-32807)/T) THEN
  PRINT#4
5070 NEXT : CLOSE4
```

The extra little bit at the end of 5040 takes care of "@" signs that have a screen code of zero. Lines 5020 and 5030 generate a RVS ON or RVS OFF character and then decide whether to send it or not (5030). This is dependent on the field of the last character. The routine has one bug though; if quotes are printed to the printer, any RVS or RVS OFF characters sent will appear literally. If you expect there may be quotes, you'll need to modify so that a CHR$(141); is sent to do a carriage return with no line feed. Then you'll need to position back to where you left off.

This routine is dreadfully slow but it was meant to be an exercise in boolean algebra.

PET Quickie

Using >$ followed by any two characters you can quickly get blocks free & disk name.

```
>$XX
 0 [illegible]
 75 BLOCKS FREE.
 1 [illegible]
 101 BLOCKS FREE.

@$10
 1 [illegible]
 101 BLOCKS FREE.
 0 [illegible]
 75 BLOCKS FREE.

>$55
 0 [illegible]
 75 BLOCKS FREE.
 1 [illegible]
 101 BLOCKS FREE.

(USES DOS-SUPPORT WEDGE)
```

ON GOTO ELSE

A useful sequence in BASIC is IF — THEN — ELSE. Unfortunately, as everyone knows, PET BASIC can't do an IF THEN ELSE, or can it? Well, the answer is no, it can't. However, a lot of times the 'THEN' keyword is followed by a line number which really means GOTO. IF GOTO ELSE we can do!

The common approach in BASIC is an IF statement followed by some 'if criteria,' followed by THEN or GOTO and a line number. Anything beyond here will be ignored since if the condition is satisfied, the GOTO is executed, and if not, BASIC drops down to the next line of the program. Therefore, the next line usually contains the 'ELSE' code. For example:

```
100 IF X=B GOTO 120 : X=X+1 : GOTO 100
```

. . . will never work. Even though it does absolutely nothing, to do it correctly we need:

```
100 IF X=B GOTO 120
110 X=X+1 : GOTO 100
```

To get it all on one line we use the ON GOTO statement:

```
100 ON -(X=B) GOTO 120 : X=X+1 : GOTO 100
```

True/false logic in PET BASIC produces a "0" for false and "−1" for true (try PRINT 4=5, 5=5). A negative argument will give ?ILLEGAL QUANTITY ERROR so we change the sign with −(A=B). Fortunately, an argument that is out of range (ie.'0') for ON GOTO will not cause execution to drop to the next line but rather continue with the next statement. This is also true if there are not enough line numbers following the GOTO to satisfy the argument. For example:

```
100 ON PEEK(32768+X) GOTO 120 : X=X+1
  :GOTO 100
```

. . . will GOTO 120 only if the character PEEKed is an "A". C=

Commodore Computers Bloom with CP/M®, Hard Disk and Multi-User Capability.

COMMODORE DEALERS

Expand your customer base by including the vast range of CP/M software development and applications!

Small Systems Engineering provides everything you need to enter the CP/M marketplace. Excellent dealer margins. Advertising and marketing support. CP/M software on Commodore formats. Corvus products. Watch for innovative products from SSE to support your Commodore Computer line.

Now your Commodore can blossom into a full-blown CP/M-based or multi-user business system with the addition of one of these innovative peripherals. Your Commodore plus the SOFTBOX gives you a single-user CP/M system. Commodore plus HARDBOX puts Pet software on Corvus Disk Drives and multiplexing units.

RUN CP/M WITH SOFTBOX ON FLOPPY AND HARD DISK.

Add SOFTBOX. The Z80-based computer which connects to your Commodore PET or CBM through the IEEE bus.

Now you have the ability to run any CP/M software or application, giving you a universe of computing capabilities you could only wish for until now.

Simply by plugging the SMALL SYSTEMS SOFTBOX into your Commodore Computer's IEEE-488 port and loading the CP/M disk, your computer will run under the world's most popular disk operating system. No internal connections or modifications to your computer are necessary.

Applications packages designed to work with specific terminals (like Lear Siegler ADM3A, Televideo 912 or Hazeltine 1500) need no modification to work with your computer's screen, since the SOFTBOX allows your computer to emulate any of these devices.

Interfaces that come with your SOFTBOX let you add the Corvus Hard Disk and any RS-232C peripheral. You can run a printer, and even talk to another CP/M based system.

90-Day Warranty. Suggested retail $895
Complete line of CP/M software available.

HARDBOX GIVES PET DOS MULTI-USER AND HARD DISK CAPABILITY.

The HardBox introduces your PET software to the world of high-capacity, high speed disk storage. It acts as an intelligent controller for up to four Corvus Winchester disk drives. Each Corvus drive can have a capacity of 6, 10 or 20 million bytes.

The HardBox is designed to be software-compatible with PET DOS version 1 or 2, so that it will work with existing programs. The HardBox is designed to appear to the PET as a fast, high-capacity floppy disk unit.

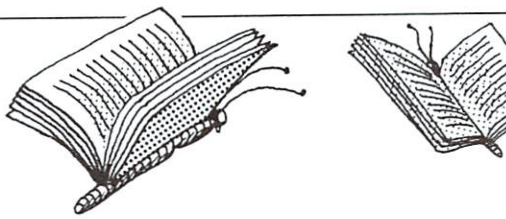
When used with the Corvus range of drives, the HardBox provides much more than just a hard disk interface. By using an 8-way multiplexing unit, up to eight PET users, each equipped with a HardBox, may simultaneously address the same hard disk. By using two levels of multiplexing, up to 64 users may be accommodated. The hard disk may be divided logically into user areas.

A comprehensive set of utility programs are provided with the hardbox for file transfer, hard disk configuration, mirror backup, system maintenance etc.

90-Day Warranty. Suggested retail $695.

SSE

SMALL SYSTEMS ENGINEERING
222 B View Street
Mountain View, CA 94041
(415) 964-8201



Book Review

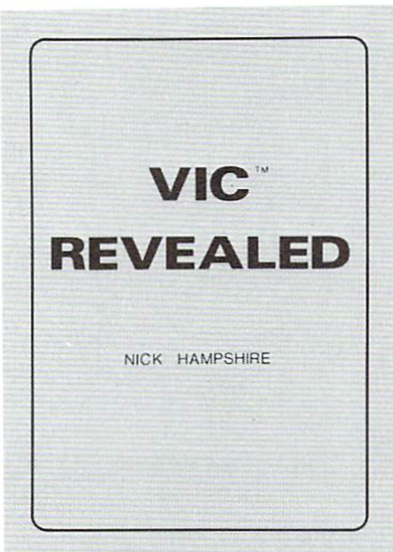
The VIC Revealed by Nick Hampshire

by
Neil Harris

Where would the personal computer industry be without people like Nick Hampshire? Manufacturers are hard-pressed to document precisely every facet of their machines, especially when the machines are new. Systems are often released with only preliminary manuals. What's the poor defenseless owner to do? That's where the Nick Hampshires of the world come to the rescue.

Nick Hampshire Publications in England released several books to help PET owners, including **The PET Revealed** and his excellent volume on **PET Graphics** (which may be of interest to VIC owners as well). Both these books are published in the USA by Hayden Books. His latest effort, **The VIC Revealed,** is only available in England so far.

This book isn't for the timid—it faces the VIC from the point of view of the serious machine language programmers/engineers, the types who re-write game cartridges and build their own interfaces to control household appliances. As such, it can be (and is) considerably more rigorous in its approach than Commodore's own **Programmers Reference Guide** (PRG) which has a much more general audience.

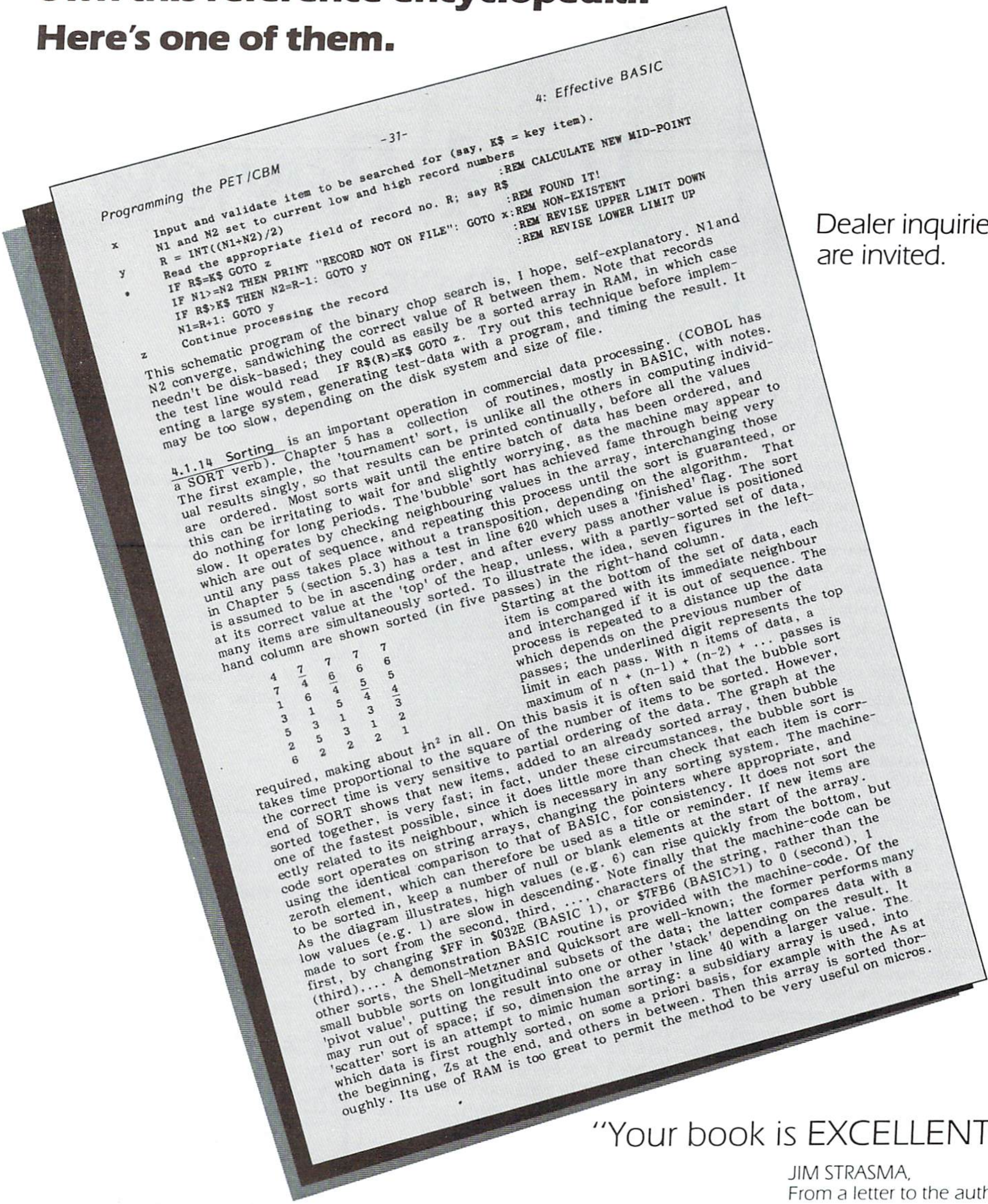


The VIC Revealed starts with the 6502 microprocessor chip and works its way outward through the ROMs, Video Interface Chip, and input/output functions. The ROM section includes a description and guide to using the BASIC ROM, which *couldn't* be in the PRG because of Commodore's original agreement with MicroSoft. It also documents the technical guide of the cassette recorder better than any other source I've seen—so good that I recommend it to companies wanting to duplicate Commodore-format tapes.

The only bad news is the redundancy of material here with that in the PRG. The opening section of **The VIC Revealed** covers the 6502 chip, which is explained in greater detail in the PRG, and even more so in specialized 6502 books like **6502 Software Design.** The sections on the Kernal ROM, V.I.C. chip and I/O are very similar to that in the PRG.

So, for you software engineers who want your VIC to do *anything* you want, and for those of you who need to know what every little wave and bit does inside the machine, here is another must for your book shelf. C=

There are over 500 reasons to own this reference encyclopedia.
Here's one of them.


Programming the PET/CBM -31- 4: Effective BASIC

```
x   Input and validate item to be searched for (say, K$ = key item).
    N1 and N2 set to current low and high record numbers
y   R = INT((N1+N2)/2)                              :REM CALCULATE NEW MID-POINT
    Read the appropriate field of record no. R; say R$
•   IF R$=K$ GOTO z                                 :REM FOUND IT!
    IF N1>=N2 THEN PRINT "RECORD NOT ON FILE": GOTO x:REM NON-EXISTENT
    IF R$>K$ THEN N2=R-1: GOTO y                    :REM REVISE UPPER LIMIT DOWN
    N1=R+1: GOTO y                                  :REM REVISE LOWER LIMIT UP
z   Continue processing the record
```

This schematic program of the binary chop search is, I hope, self-explanatory. N1 and N2 converge, sandwiching the correct value of R between them. Note that records needn't be disk-based; they could as easily be a sorted array in RAM, in which case the test line would read IF R$(R)=K$ GOTO z. Try out this technique before implementing a large system, generating test-data with a program, and timing the result. It may be too slow, depending on the disk system and size of file.

4.1.14 Sorting is an important operation in commercial data processing. (COBOL has
a SORT verb). Chapter 5 has a collection of routines, mostly in BASIC, with notes.
The first example, the 'tournament' sort, is unlike all the others in computing individ-
ual results singly, so that results can be printed continually, before all the values
are ordered. Most sorts wait until the entire batch of data has been ordered, and
this can be irritating to wait for and slightly worrying, as the machine may appear to
do nothing for long periods. The 'bubble' sort has achieved fame through being very
slow. It operates by checking neighbouring values in the array, interchanging those
which are out of sequence, and repeating this process until the sort is guaranteed, or
until any pass takes place without a transposition, depending on the algorithm. That
in Chapter 5 (section 5.3) has a test in line 620 which uses a 'finished' flag. The sort
is assumed to be in ascending order, and after every pass another value is positioned
at its correct value at the 'top' of the heap, unless, with a partly-sorted set of data,
many items are simultaneously sorted. To illustrate the idea, seven figures in the left-
hand column are shown sorted (in five passes) in the right-hand column.

				7
		7	7	6
4	7	6	6	5
7	4	4	5	4
1	6	5	4	3
3	1	1	3	2
5	3	3	1	1
2	5	2	2	
6	2			

Starting at the bottom of the set of data, each
item is compared with its immediate neighbour
and interchanged if it is out of sequence. The
process is repeated to a distance up the data
which depends on the previous number of
passes; the underlined digit represents the top
limit in each pass. With n items of data, a
maximum of n + (n-1) + (n-2) + ... passes is
required, making about ½n² in all. On this basis it is often said that the bubble sort
takes time proportional to the square of the number of items to be sorted. However,
the correct time is very sensitive to partial ordering of the data. The graph at the
end of SORT shows that new items, added to an already sorted array, then bubble
sorted together, is very fast; in fact, under these circumstances, the bubble sort is
one of the fastest possible, since it does little more than check that each item is corr-
ectly related to its neighbour, which is necessary in any sorting system. The machine-
code sort operates on string arrays, changing the pointers where appropriate, and
using the identical comparison to that of BASIC, for consistency. It does not sort the
zeroth element, which can therefore be used as a title or reminder. If new items are
to be sorted in, keep a number of null or blank elements at the start of the array.
As the diagram illustrates, high values (e.g. 6) can rise quickly from the bottom, but
low values (e.g. 1) are slow in descending. Note finally that the machine-code can be
made to sort from the second, third, ..., characters of the string, rather than the
first, by changing $FF in $032E (BASIC 1), or $7FB6 (BASIC>1) to 0 (second), 1
(third),... A demonstration BASIC routine is provided with the machine-code. Of the
other sorts, the Shell-Metzner and Quicksort are well-known; the former performs many
small bubble sorts on longitudinal subsets of the data; the latter compares data with a
'pivot value', putting the result into one or other 'stack' depending on the result. It
may run out of space; if so, dimension the array in line 40 with a larger value. The
'scatter' sort is an attempt to mimic human sorting: a subsidiary array is used, into
which data is first roughly sorted, on some a priori basis, for example with the As at
the beginning, Zs at the end, and others in between. Then this array is sorted thor-
oughly. Its use of RAM is too great to permit the method to be very useful on micros.

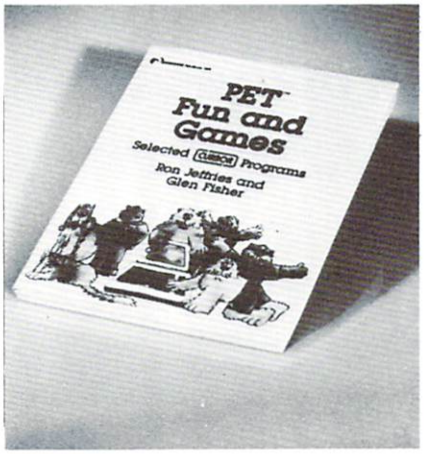
Dealer inquiries are invited.

"Your book is EXCELLENT!"

JIM STRASMA,
From a letter to the author.

A **COMPUTE! Books** Publication
In conjunction with Level Limited.

NEW PRODUCT DEVELOPMENTS



Company:
A. Osborne/McGraw Hill
630 Bancroft Way
Berkeley, CA 94710
(415) 548-2805

Product:
PET Fun and Games—Challenge friends or outwit the PET™ in games of action, risk or chance which allow you to develop your memory, your cunning . . . your strategy. Save a stranded expedition in outer space, fight monsters over gold, zap targets, climb Mt. Everest, draw faces. Thirty-one games and puzzles, which can be played on Commodore PET™ and CBM™ computers, are packed into **PET™ Fun and Games** by Ron Jeffries and Glen Fisher.

Each game is accompanied by play instructions and complete BASIC program listings to activate your computer. Knowledge of programming is not required. The selection includes games which make use of the special graphics and CB2 sound capabilities of Commodore computers.

Price:
$10 (192 pages)

Company:
Abacus Software
P.O. Box 7211
Grand Rapids, MI 49510
(616) 241-5510

Product:
GRAPHVICS—ABACUS Software announces the immediate availability of a graphics package for the VIC-20 called GRAPHVICS which provides the VIC user both HiResolution and Multicolor display modes.

GRAPHVICS gives you two screens—one for normal text and the other for graphics display. The function keys allow you to switch back and forth between the two screens. On the graphics screen you have control over 24,000 individual points. You can mix both HiRes and Multicolor modes on the same screen to create graphic pictures.

GRAPHVICS adds 18 commands to VIC BASIC—commands to set colors, plot points, draw lines and rectangles, and even display text on the graphics screen.

GRAPHVICS is interactive allowing you to type the commands at the keyboard and watch the graphics appear on the screen.

GRAPHVICS will run on any VIC that has either a 3K or 8K expander. It comes on cassette or diskette with the user's manual and sample programs.

Price:
$25 (U.S. and Canada)
$50 (elsewhere)

Company:
Cimarron
666 Baker Street, Suite 309
Costa Mesa, CA 92626
(714) 641-1156

Product:
DAS (Dental Accounting System), OAS (Orthodontic Accounting System) and VAS (Veterinary Accounting System)—DAS, OAS and VAS proceduralizes daily operations by allowing the operator to process all charges and payments through "ticket entry" programs by patient, client, animal or guarantor. Diagnostic and procedure codes are maintained as well as industry specific information.

All three programs share Cimarron's Genetic Software design which, unlike CP/M programs, allows the operator to take maximum advantage of Commodore's hardware features. For example, entire lines or screens can be scrubbed with just one key stroke. Record retrievals on partial names take less than a second and each file can be "browsed" through for ultimate operator convenience.

DAS handles standard and California procedure codes and insurance forms while maintaining treatment plans, morbidity indices and prophylaxis/next visit patient scheduling. Pre-determination estimates are also generated. Revenue reports help evaluate the cash position of the practice including 90 day guarantor aging and payment history by insurance company.

OAS incorporates the above features and is supplemented by installment billing by treatment contract and split billing for the convenience of separated guarantors. Billing by procedure code is also supported.

VAS tracks both animals and clients. Pet/animal name, species, and registration are stored in the master file with vaccination history, date of birth, weight and rabies information. Service fees are billed at time of service. VAS prints useful analysis reports in addition to the standard recall, aging and billing print outs.

Each package is priced separately at $595.00 and includes complete documentation and support materials.

Price:
$595 per package

Company:
College of Education
University of Oregon
Eugene, OR 97402
(503) 686-3405

Product:
The Computer: Extension of the human mind—As an educator, how do you view computers? Are they a complex confusion looming on your professional horizon, or do you think of them as functional tools with which to enhance teaching and learning?

Whatever your view, the College of Education, University of Oregon, offers an opportunity to learn more about adding computer literacy to the three Rs at its third annual Summer Conference, "The Computer: Extension of the Human Mind," July 21-23, 1982, at the Eugene Hilton Hotel and conference center. National leaders in the field of computer science and in computer manufacturing will explore the current state of computers in education and provide a glimpse of future trends. A variety of computers will be displayed and operated and both general and small group sessions will focus on topics germaine to adding computer technology to education.

Registration and program details may be obtained by writing: '82 Summer Conference, Jude Ridge, College of Education, University of Oregon, Eugene, OR 97402 (503) 686-3405.

Price:
N/A

Company:
DP Directory
P.O. Box 562
Bloomfield, CT 06002

Product:
D P Directory—DP Directory, a new data processing reference magazine, publishes the tables of contents of over 100 DP periodicals each month. DP Directory covers dozens of data processing magazines dealing with hardware, software, systems development, telecommunications, graphics, word processing and personal computing. 12 monthly issues are available for $48 from DP Directory, P.O. Box 562, Bloomfield, CT 06002.

Price:
See Description

Company:
Electronic Specialists
171 South Main Street
Natick, MA 01760
(617) 655-1532

Product:
Interference Control and Electronics Products Catalog—A new 40-page catalog from ELECTRIC SPECIALISTS presents their line of MICROCOMPUTER interference control products. Protective devices, Line Voltage Regulators and AC Power Interruptors are also included.

Descriptive sections are included outlining particular problems and suggested solutions. Typical applications and uses are highlighted. Request Catalog 821.

Price:
N/A

Company:
Hayden Book Company
50 Essex Street
Rochelle Park, NJ 07662
(800) 631-0856

Product:
I Speak BASIC: A Computer Literacy Package—I Speak BASIC is a machine specific computer literacy course that introduces students to BASIC programming. The course provides student instruction in the BASIC language for the Apple, TRS-80 and PET and includes a Teacher's Manual, Student Text and Exam Set for each machine.

Written by Aubrey Jones and published by the Hayden Book Company of Rochelle Park, New Jersey, I Speak BASIC is designed for teachers regardless of their knowledge of microcomputers and their programming skill.

The core of the course is the Student Text that features learning objectives, definitions and examples of key terms and BASIC concepts, in class programming exercises, practices and assignments. Each version includes chapters explaining the parts and operation of the microcomputer. Chapters cover BASIC programming topics such as Mathematical Operations, Scientific Notations, Conditional and Unconditional Branching, Input Statements, Loops, Reading Data, Video Display Graphics, Arrays and Subroutines.

The Teacher's Manual provides techniques for presenting the material and emphasizing particular concepts, annotations to aid in lesson planning, suggestions for implementing the course and answers to all practice exams.

The Exam Set contains 12 quizzes on spirit duplicating masters to check student understanding and reinforce learning. The quizzes can be easily reproduced for class use. A classroom set of I Speak BASIC contains one Teacher's

Manual, 20 Student Texts and one Exam Set.

Price:
N/A

Company:
Reston Publishing Co.
11480 Sunset Hills Road
Reston, VA 22090
(800) 336-0338

Product:
PET BASIC—Reston Publishing Company has announced the publication of a new computer programming book, PET BASIC, by Ramon Zamora, William Scarvie and Bob Albrecht.

Perfect for the beginning PET user, this book is filled with lively examples, do-it-yourself exercises and creative explorations.

Experienced PET users will find this book a handy reference guide. You'll discover a variety of things your PET can do. Each new piece of information is presented logically, step-by-step with open page formats and lots of humor.

PET BASIC can also be used to teach children to use the PET (age 9 and up). You'll find games and exercises and current symbols in children's world characters.

Price:
$14.95 (paperback)
$19.95 (cloth)

Company:
Reston Publishing Co.
11480 Sunset Hills Road
Reston, VA 22090
(800) 336-0338

Product:
PET Games and Recreation—Reston Publishing Company has announced the publication of a new computer games book, PET GAMES AND RECREATIONS, by Mac Oglesby, Len Lindsay, and Dorothy B. Kunkin.

Start at the beginning, or jump in somewhere in the middle! "Plan Ahead Games" tests your ability to think out winning strategies. "Deductive Reasoning Games" places you in the role of detective—you must use the information given to work out the solution. Combine logic with good fortune as you gamble against the computer using "Games of Chance." "Words and Numbers" develops counting and language skills. "Recreations" provides pure entertainment. Just follow the playmaster's instructions and sit back and watch your PET computer bounce balls and do all kinds of wild things!

You'll love Professor Whacko's Lecture Series on Computer Whacko Science. Dr. Whacko introduces many of the mysteries of PEEKS, POKES, and Subroutines, stimulating you to experiment and explore.

Price:
$12.95 (paperback)
$17.95 (cloth)

Company:
Small Systems Engineering
71 Park Lane
Brisbane, CA 94005
(415) 468-2900

Product:
SoftBox, HardBox, and Petspeed—Three new products that upgrade Commodore computers into powerful business and educational tools have been introduced from England by Small Systems Engineering, Inc. of Mountain View, California.

Called the SoftBox™, HardBox™, and Petspeed™ compilier, these enhancements bring the benefits of the widely used CP/M™ operating system, Winchester hard disk mass storage, multi-user capability, and high-speed BASIC compiling to all PET and CBM microcomputers.

—The Softbox permits Commodore users to run hundreds of CP/M compatible applications packages, as well as interface with up to four Corvus Winchester hard disk drives. RS232 interfacing capability is also included.

—The HardBox, teamed one to a computer, will allow up to 64 users to access simultaneously the same Corvus hard disk storage—up to 80 Mbytes using the Corvus Constellation multiplexer.

—The Petspeed compiler allows Commodore Basic programs to run up to 30 times their normal interpretive speed. The software includes optimization procedures that permit faster execution than other compilers.

The SoftBox, containing a Z80-based 64K RAM board, modifies the CP/M operating system for the Commodore disk drive, using the PET or CBM computer itself as a terminal. CP/M version 2.2 software is included, and runs at 4MHz with no wait states, for rapid execution.

A proprietary SoftBox system utility called NEWSYS gives users much latitude in reconfiguring the operating system for their own requirements. Menu-driven options include disk drive, I/O, and RS232 assignment, as well as allowing the computer to emulate a Lear Siegler ADM3A, Televideo 912, or Hazeltine 1500 terminal.

Small System's HardBox device enhances the PET disk operating system (PET DOS versions 1 or 2), allowing one to four Corvus drives to emulate the Commodore floppy disk unit for up to 64 users.

Seven HardBox utilities are also included: user reconfiguration, password security, file transfer between hard disk and floppies, diagnostics, and use of a video cassette as a backup device.

The Petspeed compiler uses a unique four-pass algorithm that gives priority to frequently used variables, removes unnecessary code, and utilizes integer arithmetic wherever possible. In addition to its fast execution, the compiler is available with unrestricted use. Software writers may sell their compiled programs without incurring additional royalties.

Price:
$895 (HardBox & SoftBox)
$350 (Petspeed)

Complier Comments

by
Jim Butterfield, Toronto

I don't want to become involved in the Great Debate about compilers. On the other hand, it's almost irresistable to dive in and add a few footnotes. You'll find no product reviews here. Just a little talk about what's involved.

For BASIC?
Some languages were designed for compilers. In fact, the compiler was designed first, and whatever it turned out you had to type in ended up as the language. FORTRAN started more or less this way. To put compilers in perspective, we have to do a little historical work.

Once, long ago, there were no interactive computers. You punched up a deck of cards and if you were lucky an operator would run them sometime that week. Most of the results came back saying something like SYNTAX ERROR (does that sound familiar?). There was no point in having an interpreter language; you wouldn't be there to watch it happen. We had FORTRAN and COBOL and others.

The first FORTRANs, for example, were tricky. If you used a variable called DIGIT, it would turn out to be a floating-point number; on the other hand a variable called NUMBER would be fixed-point. Heaven help you if you typed TOTAL = TOTAL + 1; you'd get a ?MIXED MODE error notice and have to recode TOTAL = TOTAL + 1.0 to fix it. To input or output you needed to give more than the command: an extra line called FORMAT was needed, written in advanced gibberish. Honest.

Many of these problems have been corrected over the years—you did know that there was more than one FORTRAN, didn't you?—but the style remains. The programmers have to adapt to the machine, and interactive is still an alien concept.

And Now, BASIC . . .
Along came BASIC. It's a loose language: you don't have to dimension some arrays; strings wander all over; sometimes you can have FOR and NEXT items that don't match (bad practice, but it can be done) . . . and interactive users love it.

What's the problem? Some things are not clearly defined by BASIC. Let's look at a few of them.

Strings may be the worst thing that a compiler has to deal with. BASIC doesn't tell the compiler how big any string is likely to be—ever. INPUT X$ gives no hint as to the size of string X$. The poor compiler has a grim choice: allow maximum space for all strings and waste a lot of memory; or bounce the strings around as they change. The first alternative costs you program size; you write this little program that says DIM A$(1000) and the compiler immediately reports OUT OF MEMORY since it tries to allocate 255000 bytes for the array. The second alternative costs you time; no matter what you call it, some sort of garbage collection will have to take place. And then people complain because they expect compilers to produce fast fast code.

At first glance we think that the whole object of compiling is to get speed. But we don't give the compiler enough information to work up a really fast program. It's obvious that FOR J = 1 TO 10 can run faster if we treat J as an integer. Unfortunately, we're not allowed to code FOR J% so the compiler will have to figure it out for itself. And what will it do with FOR J = A TO B? Until A is computed, we cannot know if it's integer or not.

It's obvious to us. We wrote the program. But the dumb compiler can't read our minds; and BASIC doesn't give enough explicit information to do the job.

One last example. It's one of the annoying things about BASIC that we sometimes have to code things like GET#1,X$: IF X$ = "" THEN X$ = CHR$(0) mostly to cover failings in BASIC itself. If I were hand-coding into machine language, I could replace the whole thing with one instruction, because I know that Machine Language doesn't have the "fault" that's in BASIC. But a poor compiler can't know that. It sees the GET instruction and codes it . . . and it must add to the coding to generate the BASIC "fault" if it wants to be compatible. Then it must proceed to the IF statement and work through the coding to fix that same fault.

The Choices.
The compiler designer has a choice. He can code for 99% compatibility, tracking everything that BASIC does quite exactly (including the faults). In doing so, he'll create a package in which almost anything will compile successfully. But, the compiled machine language will be doing most of the things that BASIC does, and won't be much faster than BASIC.

On the other hand, the designer can ask the user to make changes to his program before compilation that will help the process. He may also have things that compile from BASIC in a non-standard manner. He may make arbitrary decisions on BASIC structures—all FOR loop variables will be fixed-point, for example. And the compiler may question the user during compilation: How large is string M$ likely to be? Can J be fixed-point? The user has to work harder, but the end product runs faster.

Either way, the compiled program is not likely to be smaller in size than its BASIC source. It's difficult to code 100 IFJ>5THENPRINT"J IS";J in less than the 19 bytes that BASIC uses. And good compilers add

extra arithmetic—fixed-point addition, for example—that takes up overhead space.

Why Compile?
It's your choice. If you have a program that runs for five hours, you will probably be delighted with a paltry four-to-one compiler speedup. If you want protection against listing, a compiler will do a good job of instant confusion.

Don't lose perspective. A program that spends most of its time waiting for an operator or for a printer won't speed up much under compilation.

Machine Language Programmers will be happy to know that they are not yet obsolete. Compilers can do a useful job. But until they get the brains equivalent of a human's judgment, they won't replace hand coding.

Machine Language Auto-Location

When a program like Supermon or Tinymon loads into its computer and RUN is given, it builds a copy of the "real" program in high memory. There's a need to do this: different computers have different memory sizes, and we want to find the top of memory wherever it is. More: the computer might already have something else near the top of memory (such as a wedge program) and we want the new program to fit neatly below it.

This calls for an auto-location program. The object program must be packed into high memory. This is often more than just moving the program, since some things may need to be changed with the move. If you have a program that uses only branches—no jumps, no in-program subroutines, no tables—you may be able to get away with a simple move operation. But any instruction that uses an in-program absolute address: jumps, subroutine calls, and tables—will need to be adjusted.

We need to build a relocatable program module. Something that says, "This byte is normal so we may just move it; but that pair of bytes is an address and must be recalculated for the new location."

Ground Rules.
We need a scheme which marks addresses so that the proper arithmetic may be performed. There's one requirement as to how you write the program: it may be summarized as "all addresses must be in one piece". . .

The rule makes sense: it would be difficult to perform arithmetic on an address whose two bytes were scattered in different parts of the program. For users with assemblers, the rule translates to: never use the < or > functions for high and low byte.

So if we wanted to place the address of TABLE into indirect address INDAD, we would avoid coding: LDA #<TABLE : STA INDAD : LDA #>TABLE : STA INDAD+1. Instead, we'd define the table address in memory with TABLAD .WORD TABLE and perform the above setup with LDA TABLAD : STA INDAD : LDA TABLAD+1 : STA INDAD+1. We've used four more bytes but gained a major benefit: the two bytes representing the address of TABLE are now stored together (at TABLAD) and we can adjust this address easily when we wish to relocate.

The Method.
The way we build a relocatable module is quite easy. Any time we see an address that will need relocation, we place a zero above it. As we repack the program (from the top down) the zero will signal that a relocatable address follows. That's all very well, but what do we do with real zeros? There will be many zeros in the program itself, and we don't want them to trigger a false relocation calculation. In this case, we change the zero to two zeros in the relocatable package. The relocation program will spot this and change it back to a single zero.

In order to do arithmetic on the addresses, we need to know where they are pointed in the first place. To relocate from $1000 to $4000, for example, we need to add $3000; but we must know that we are starting from $1000. I use the following convention: addresses are written so that the top of the program plus one is $0000—that is, the last byte of the relocatable program is $FFFF. The program can't really go there, since that's ROM space, but it makes the arithmetic easy. We can look at an address in the relocation package as a signed number: address $FFC0 can be viewed as "64 bytes from the top of the program." If our real top-of-program turned out to be $8000, which would be correct for a 32K machine, we would translate the sequence 20 C0 FF 00 to 20 C0 7F . . . note that the zero disappears; it's the relocation flag. How did we get the new address $7FC0? By adding the relocation address, $FFC0, to the top-of-program, $8000.

Generating the Relocatable Program.
How do we manufacture this package with zeros added and addresses recalculated, ready for relocation? With an assembler it's quite easy.

First, we assemble two versions of the program at two different locations. That's easy enough to do: we just change the * = statement at the start of our source code.

Then we run a simple compare program which compares the two object programs we have assembled, starting from the top. Each matching byte is copied into the relocation area unchanged; if it's a zero, an extra zero is added. If the bytes don't match, we have a relocatable address: in this case, we insert the zero plus the recalculated address into the relocation package. It's an easy job: my "relocate builder" is a BASIC program of about a dozen lines.

Stopping.
As we work down from the top we need to detect when we have reached the end of the program: this is true of both the relocate builder and the relocating program itself. There are many easy ways of doing it. The program can test to see if the last address has been reached. Alternatively, we can put some sort of "flag" into the coding itself to detect the end. In TINYMON, I use a value $BF which is never used in the program as a simple detection. A more complete method might be to use a zero with a value of 1 stored below it. It's up to you: whatever works is OK.

VIC Note.
In the VIC, we have one more problem to solve. We can find the top of memory (locations $37 and $38) but our program might fall into different memory space, depending on what's plugged in. Use pointers to find your own program (try $2D and $2E) and everything should work out nicely.

Summary.
You can pick apart the code of SUPERMON or TINYMON and see how it's done. You can develop your own programs. But if you understand the principles of a relocating program package, you can develop significantly more useful programs which will adapt to a wider variety of machine configurations.

Editor's Note
The machine code disassembly to follow is Jim Butterfield's relocator modified slightly by Dave Hook for use with his Vicloader for PET/CBMs. Dave eliminated the JMPs and JSRs in Jim's original utility so that the relocator can be relocated. For Vicloader, it starts at $0640, but you can move it anywhere; higher if you want more BASIC underneath it, or lower for larger object programs.

Notice that the relocator starts with the end of the object program since this will be the first byte to be packed into high memory. This is conveniently pointed at by the Start of Variables pointer minus 1, which is set on completion of the LOAD (provided it is .Saved properly).

```
0400-063F   BASIC portion (title, sys address, etc)

0640 A5 2A     LDA $2A       ;store copy of
0642 85 1F     STA $1F       ;Start of Variables
0644 A5 2B     LDA $2B       ;pointer (last byte of
0646 85 20     STA $20       ;object program + 1).
0648 A5 34     LDA $34       ;store copy of
064A 85 21     STA $21       ;Top of Memory
064C A5 35     LDA #35       ;pointer (MemTop)
064E 85 22     STA $22
0650 A0 00     LDY #$00      ;zeroise Y index
0652 A5 1F     LDA $1F       ;dec pointer to last
0654 D0 02     BNE $0658     ;byte of object prog.
0656 C6 20     DEC $20       ;(lst byte to be
0658 C6 1F     DEC $1F       ;packed)
065A B1 1F     LDA ($1F),Y   ;get obj. prog. byte
065C D0 3C     BNE $069A     ;not 0, goto $069A
065E A5 1F     LDA $1F       ;if 0, dec pointer
0660 D0 02     BNE $0664
```

```
0662 C6 20      DEC $20
0664 C6 1F      DEC $1F
0666 B1 1F      LDA ($1F),Y    ;and get next byte
0668 F0 21      BEQ $068B      ;0? yes, true zero *
066A 85 23      STA $23        ;no, relocatable addr
066C A5 1F      LDA $1F        ;store high byte in
066E D0 02      BNE $0672      ;$23.  dec pointer
0670 C6 20      DEC $20        ;and
0672 C6 1F      DEC $1F
0674 B1 1F      LDA ($1F),Y    ;get next byte
0676 18         CLC            ;recalculate lo addr
0677 65 21      ADC $21        ;using MemTop lo
0679 AA         TAX            ;result in .X
067A A5 23      LDA $23        ;recalculate hi addr
067C 65 22      ADC $22        ;using MemTop hi
067E 48         PHA            ;result on stack
067F A5 34      LDA $34        ;dec MemTop
0681 D0 02      BNE $0685
0683 C6 35      DEC $35
0685 C6 34      DEC $34
0687 68         PLA            ;retrieve hi addr
0688 91 34      STA ($34),Y    ;pack at ($Memtop) .Y=0
068A 8A         TXA            ;retrieve lo addr
068B 48         PHA            ;* save on stack
068C A5 34      LDA $34        ;dec MemTop
068E D0 02      BNE $0692
0690 C6 35      DEC $35
0692 C6 34      DEC $34
0694 68         PLA            ;retrieve byte
0695 91 34      STA ($34),Y    ;pack at ($MemTop) .Y=0
0697 18         CLC            ;rather than
0698 90 B6      BCC $0650      ;a JMP
069A C9 BF      CMP #$BF       ;last byte?
069C D0 ED      BNE $068B      ;no, goto $068B *
069E A5 34      LDA $34        ;yes, set
06A0 85 30      STA $30        ;Bottom of Strings
06A2 A5 35      LDA $35        ;= MemTop
06A4 85 31      STA $31        ;pointer
06A6 6C 34 00   JMP ($0034)    ;jmp to program
06A9 BF                        ;end detector of obj prog
06AA ...                       ;start of object prog
```

Getting the Most out of Her Commodore Computer

In the Bitdiddiling section we try to find people who use Commodore computers in different and interesting ways. Using an 8032 to run a community sex information center sounded like something tailor-made for Bitdiddling. However, this story is actually about a special (and busy) woman.

The woman is Karen Kornhauser, who, among other things, is Director of training at the Community Sex Information Center in New York. The center provides a "hotline" for people who have questions about sex. Kornhauser started using the computer because "we had logged every call for ten years, but no one ever did anything with the information." She is using a CBM 8032, an 8050 disk drive, and Jinsam software to put 10 years of information to work.

"I compare all the data and decide which areas to stress in training" Kornhauser said. Workers at the center must go through 300 hours of training. Kornhauser is also using the data to write booklets for parents about what kids want to know most about sex.

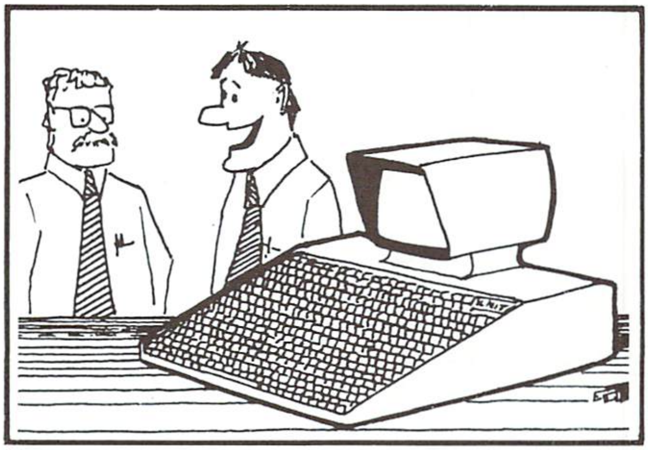
The computer keeps track of a mailing list of contributors and a list of volunteers that help keep the center running. Kornhauser is also using Jinsam and Wordpro to get volunteers to come to meetings. "The letters look just like telegrams, and we never had so many people show up for meetings," Kornhauser said.

Kornhauser uses her computer for several other things. She is using her computer to keep data for her doctorate on "how people from different cultures perceive health education messages." She is going to Sri Lanka soon for research.

But wait, there's more—she is also using her computer and a modem to get computer time on New York University's mainframe computer. She is required to do a certain amount of time on the computer for a research course. "I live in West Chester and it's difficult to get computer time but with the modem it's no problem" she explained.

Kornhauser is the dean of students at the Dodge Vocational School in New York, where she is using her computer for classroom management. She has even toyed with a program that will analyze how the phases of the moon affect student behavior. It seems that Kornhauser really knows how to use a computer to its fullest potential. But that seems only natural from someone who is living her life to the fullest.

Reprinted from *The Transactor*



"Here You See Our Entry to the Far East Market."

"I Think a Simple Error **Message** Would Suffice."

"I Said I Needed Another DISK Pack."

Pet Parking

Lon Oehlin in Peoria, Illinois is setting up a system in Chicago for parking lots using Commodore 8032s. The system will help cut down on employee theft and make it easier to enter figures into a large mainframe.

The system will count each car as it comes in and automatically print a ticket with the time it arrived. When the car leaves, the fee is calculated by the computer and appears on the screen. When the fee is paid the message "Thank You, Come Again" appears on the screen and a receipt is printed. The computer even opens the gate to let the cars leave.

This system will use one 8032 at each gate without a master control unit. If one of the computers breaks down, the others will not be affected. To save money Oehlin is using cassette drives. A large computer could do all that his system is going to do, and may even be easier to set up. Oehlin's system though, will do all the same things at one-fifth the cost.

—John O'Brien

PROJECTIONS & REFLECTIONS

I'd like to talk about one of the most exciting software products I've seen this year—Petspeed. This new software entry is a BASIC compiler designed to provide the maximum possible execution speed for programs written in Commodore BASIC. In fact, Petspeed can increase the speed of BASIC programs by even a factor of 40—No kidding!

Having used the compiler myself, I'm impressed with the way it handles my favorite utilities and applications. The compilation time is fast and the results are super. Now, all those "tricks of the trade" can be put away for keeps. With Petspeed, you can easily code for clarity, use a lot of REMs, and structure the coding so it can be easily changed later on.

Petspeed compiles a program, writes it onto disk, and leaves it in memory for immediate testing. There are utilities on the system disk for decoding compiled data, to locate variables for interfacing to assembler routines and to aid in finding "run time" errors that can occur during testing. The Petspeed compiler interfaces with our Integer Basic Compiler product, which compiles BASIC into native 6502 machine code. The Integer Basic package is great for speeding up those special IEEE routines by as much as 150 times faster. This code can then be interfaced onto the whole routine that has been compiled with Petspeed. The tandem of Petspeed and Integer Basic practically eliminates the need for coding applications for speed using Assembler.

Petspeed is a must for the serious programmer as well as the weekend "hacker," who simply enjoys fast coding. Petspeed's retail price is $300, and the product does not require any run-time protection devices for those who wish to sell compiled routines. With this newest software addition, the average coder can be more powerful than any Universal DOS Wedge, leap tall listings in a single bound, and move faster than a speeding IRQ vector. So buy, buy Petspeed, and bye, bye DTL.

—Paul Goheen
Software Product Manager

Advertisers' Index

commodore

JINSAM

Data Manager selected by NASA, Kennedy Space Center

SAVE TIME! SAVE MONEY!
JINSAM saves taxpayers 100's of thousands of $

OFFICIALLY APPROVED BY COMMODORE

"Much more powerful than you can imagine!"

Robert Baker
Kilobaud Microcomputing

JINSAM is an integrated system. It makes it easy to use your information to its fullest. No more will hundreds of valuable hours be spent searching or analyzing needed information nor re-entering information for various reports.

JINSAM transforms your desk-top computer into the "state of the art" data processing machine with features and accessories found nowhere, even at 10 times the price. NASA, Kennedy Space Center selected JINSAM 8.0 and saved approximately $95,000 over other software/hardware costs. Riley County, Kansas also selected JINSAM 8.0 and saved approximately $90,000 over other software/hardware costs.

JINSAM is designed for you. It is forgiving. It has help commands for every option, available at the touch of a button. The amount of information you store, its structure and/or your hardware can change but your data won't have to be re-entered. Recovery utilites are included even for catastrophes, security passwords are built in for privacy, simple editing and entry includes auto recall, and deleting records is easy and the space is reclaimed. JINSAM includes TWO FREE accessories for reports and labels. You have unlimited report formats with summing and lined up decimals and the label printer prints up to 5 across – any size and even allows single envelopes or index cards.

JINSAM 1.0 allows fast and easy file handling, manipulation and report generation for any CBM computer with CBM 2040 disk drive. It features a menu for ease, has encrypted passwords, 3 deep sorts, .5 to 3 second recall.

JINSAM 4.0 for CBM 4000 series adds JINSORT, user accessible machine sort of 1000 records in 15 seconds; compaction/expansion of information, automatic list maintainance, unlimited number of fields, unlimited record length and much more.

JINSAM 8.0 for CBM 8000 series has all 4.0 features plus unlimited sort, horizontal format, and search by key or record number.

JINSAM 8.2, NEW FOR '82 expands 8.0 capabilities by adding information search by word, key or record number and machine language print, format and manipulation routines.

- ★ CUSTOM DATA FILES
- ★ CUSTOM REPORTS/LABELS
- ★ KEYED RANDOM ACCESS
- ★ FAST/EASY/MENU DRIVEN
- ★ MULTIPLE SEARCH KEYS
- ★ PRIVACY ACCESS CODES
- ★ WILD CARD SEARCH

JINSAM EXECUTIVE version (soon to be released) is our most powerful professional system for the CBM 8000 and 9000 series. Executive will have 8.2 extended features plus allow multiple users with in-use lockout protection, executive command files, automatic math relations, join, merge or link files, greatly increased record capacity and machine information search by word, as well as by key or record number and many, many more features.

There are currently 7 more interfacing modules – and more under development, including independent interfaces between JINSAM and business packages for your G/L, A/R, A/P needs. We announce the availability of modules and enhancements in JINSAM's quarterly newsletter.

WORDPROPACK - Intelligent interface for WordPro 3, 3+, 4, 4+, creates lists of information from JINSAM files. It allows up to 10 conditions based on each item of information. Produce individualized letters, report cards, special reports, checks, invoices, etc.

MULTI-LABEL - Prints multiple labels per record with up to 2 lines for messages and consecutive numbering. Produce inventory, caution labels, bulk mail labels, etc.

MATHPACK - global calculator/editor +, −, *, /, by another field or constant; null (remove contents) of a field or replace contents of a field with any word, number or phrase. Sum multiple fields in each record or running sum of single field in all records. Extract information or effect permanent change. Replace in the same field or place in a waiting field.

DESCRIPTIVE STATPACK - Determine MEAN, MEDIAN, MODE, STANDARD DEVIATION, VARIANCE, RANGE. Generate HISTOGRAMS from 1 to 25 steps, and produce Z-SCORE reports.

ADVANCED STATPACK - (You must also acquire DESCRIPTIVE STATPACK) Generate one, two or three way CROSSTABS (number of occurance) CHI SQUARE, LINEAR REGRESSION, with graphic representation and prediction, LINEAR CORRELATION and SIMPLE ANALYSIS OF VARIANCE.

CALCPACK - 2 way interface to VisiCalc or any user program. It lets you use VisiCalc for complex manipulation, editing, placing results in JINSAM for sorting, storing or moving data to WordPro as well as giving the ability for exchange with your own applications.

INTERAC - Interface which can read VisiCalc files, WordPro files and almost any sequential files to build JINSAM databases automatically. For example: You could "download" information on holerith cards to sequential files and INTERAC would place them into JINSAM files.

All accessories are accessed thru the JINSAM menu and require security password to gain entrance.

JINSAM gives you FREEDOM OF CHOICE. Start with JINSAM 1.0 and upgrade hardware and data at any time. Choose from accessories at any time. The JINSAM Newsletter brings the latest updates, user input and uses and keeps an eye on the future.

JINSAM stands alone by placing "a lot of potential computing power in one integrated program package" (Fred Klein, Ferson div. of Bausch & Lomb). "The JINSAM package is justification for buying a system no matter what the hardware, be it Vector or Commodore or whatever the system" (Larry Colvin, Micro Computer Systems). It is a "grandfather" in this young field since the first JINI MICRO database system has been marketed since 1979. JINSAM EXECUTIVE will be the third generation in development. All JINSAM systems are sophisticated and flexible yet easy to use. JINSAM is saving its users valuable time and money in government, educational and research institutions, business and industry worldwide.

JINSAM is a Commodore approved product. See your local dealer for a demonstration.

JINSAM is a trademark of JINI MICRO-SYSTEMS, Inc.
WordPro is a trademark of Professional Software, Inc.
VisiCalc is a trademark of Personal Software, Inc.
CBM is a trademark of Commodore Business Machines, Inc.

JINSAM Data Manager

.... Additional Information and nearest dealer
.... Newsletter Subscription ($5 US/$8 Worldwide)
.... JINSAM Demo Disk ($15, plus shipping & tax)
.... User's Guide ($40.00, plus shipping & tax)

Please send to:

Name ________________

Position ________________

Company ________________

Address ________________

City, State, Zip ________________

Phone (   ) ________________

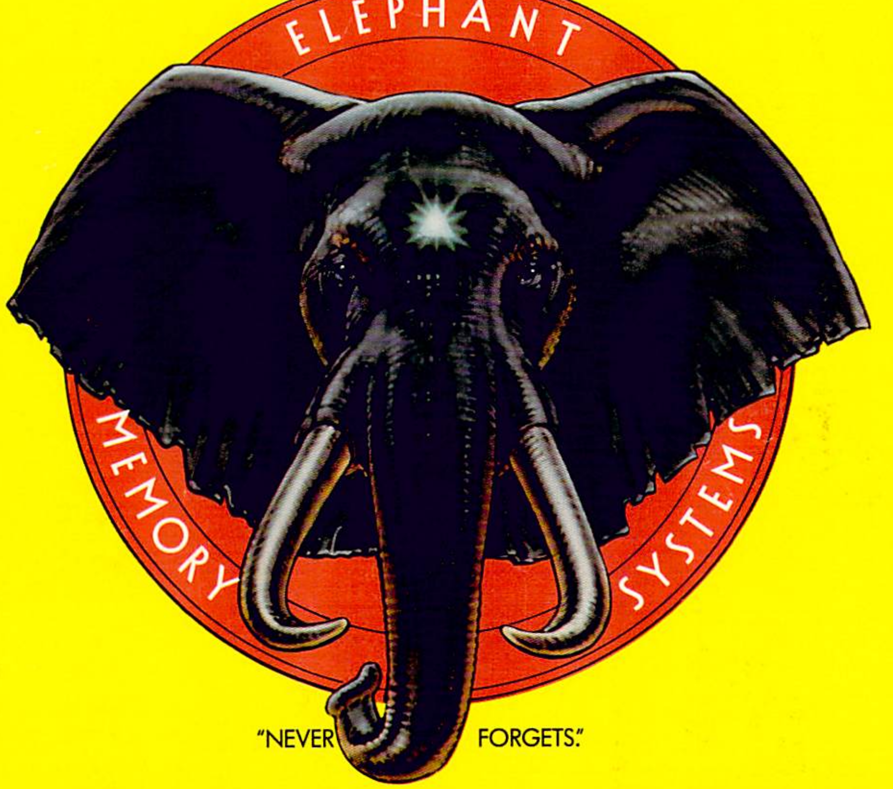
Computer, Disk ________________

Application ________________

JINI MICRO-SYSTEMS, Inc.
Box 274-CI • Kingsbridge Station
Riverdale, New York 10463
(212) 796-6200

Dealer Inquiry Welcome

REMEMBER:



MORE THAN JUST ANOTHER PRETTY FACE.

Says who? Says ANSI.

Specifically, subcommittee X3B8 of the American National Standards Institute (ANSI) says so. The fact is all Elephant™ floppies meet or exceed the specs required to meet or exceed all their standards.

But just who is "subcommittee X3B8" to issue such pronouncements?

They're a group of people representing a large, well-balanced cross section of disciplines—from academia, government agencies, and the computer industry. People from places like IBM, Hewlett-Packard, 3M, Lawrence Livermore Labs, The U.S. Department of Defense, Honeywell and The Association of Computer Programmers and Analysts. In short, it's a bunch of high-caliber nitpickers whose mission, it seems, in order to make better disks for consumers, is also to make life miserable for everyone in the disk-making business.

How? By gathering together periodically (often, one suspects, under the full moon) to concoct more and more rules to increase the quality of flexible disks. Their most recent rule book runs over 20 single-spaced pages—listing, and insisting upon—hundreds upon hundreds of standards a disk must meet in order to be blessed by ANSI. (And thereby be taken seriously by people who take disks seriously.)

In fact, if you'd like a copy of this formidable document, for free, just let us know and we'll send you one. Because once you know what it takes to make an Elephant for ANSI . . .

We think you'll want us to make some Elephants for you.

ELEPHANT.™ HEAVY DUTY DISKS.

Distributed Exclusively by **Leading Edge Products, Inc.**, 225 Turnpike Street, Canton, Massachusetts 02021
Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.